**Response to the comments**

**Associate Editor:**

This paper proposes a decomposition method for time series classification, with application in activity recognition. Its main contribution is to address the domain-generation problem. The problem is interesting and motivated by a real-case problem, and overall the paper is clearly written with sufficient experiment results. However, there are still some concerns to be addressed.

Reply to Associate Editor:

Thank you for your comments. We have addressed your concerns point-by-point as follows, and revised our manuscript according to your suggestion.

1. The current paper structure does not align well with TASE. The methodology section is too short and brief, while the experiment section is too long. I suggest writing more details about the methodology part, particularly highlighting its innovative aspects, and moving some of the experiments to the appendix.

**Reply:** Thank you for your suggestion. We have revised both method section and experiment section of the paper. In summary, we have improved the following sections:

1. Add the reason for introducing time-series decomposition in MTSDNet at the beginning of III. METHOD.
2. Explain the differences between time-series prediction tasks and time-series classification tasks (HAR) at the beginning of III. METHOD.
3. Add the effect of sliding window normalization on distribution alignment.
4. Optimize the expression of formula 4 and formula 5 for easier understanding. In the revised version, an additional formula 3 has been added. Formula 3 defines the constraint functions for both trend and seasonal term.
5. Remove detailed accuracy tables for each part. The violin plot is preserved. All related experiments have been modified.

2. The rationale behind such a simple decomposition method's ability to solve the domain-generation problem should be clearly stated. This can provide deeper insights and better demonstrate the scientific significance of the proposed method.

**Reply:** Thanks for suggestions. We have added the rationale explanation at the beginning of III. METHOD. MTSDNet is able to generalize across domains from two aspects: feature decoupling and distribution alignment, where time-series decomposition is used for decoupling and sliding window normalization is used for distribution alignment. The final features from time-series decomposition are simplified and decoupled, which helps reveal hidden knowledge in domains and improves generalization ability. Sliding window normalization ensures that the features extracted by time-series decomposition belong to the same distribution. ~~Both techniques are commonly used in domain generalization methods.~~

Especially in the HAR task, time-series decomposition is capable of distinguishing the differences between various human behaviors so that it enables MTSDNet to transform complex and coupled features into simple ones through trend, seasonal and general terms. Therefore, MTSDNet is able to simplify complex domain generalization problems, thus having advantages in accuracy and model size. We have added the detailed explanation to the main text, as follows:

On Page 3:

*“Domain differences in HAR are different from those in image fields. Complex domain differences can be regarded as the superposition of the above-mentioned errors. A simple idea is to deal with these differences separately. MTSDNet uses time series decomposition to decouple errors and features. Time series decomposition can decouple the domain differences and simplify the problem.*

*Time series decomposition are often used for time series prediction tasks such as predicting geographic and financial data. In these tasks, different components are discarded or retained because of different task concerns. The task that focus on long-term trends needs to eliminate the effects of seasonal terms and residuals, vice versa. Time series decomposition can eliminate the interaction between different components.*

*Time series decomposition requires determining which components to construct. In HAR task, time series decomposition in MTSDNet focuses on trend term, seasonal term and general term.*

*The data bias can be captured by the trend item, the periodicity of human activity can be captured by the seasonal item, and the remaining part forms the general term and residual.*

*Here general term is set up to capture sporadic signals. Sporadic signals are often ignored as residual in time series prediction tasks, but in HAR tasks, it is highly correlated with falling.*

*By using time series decomposition, Specific domain differences will be limited to a certain component. For example, the data bias caused by wearing is only generated in the trend item. When there is only data bias, temporal decomposition can ensure that the seasonal and general terms are not disturbed. Even if the classification of trend term is incorrect due to domain differences, seasonal and general terms can be correctly classified. Therefore, time series decomposition can enhance the domain generalization ability in HAR tasks.*

*What's more, time series decomposition is helpful to explicit feature alignment. Explicit feature alignment explicitly makes the feature distributions of multiple source domains as close as possible. By learning a decomposition, each layer of components is located in the same feature space. Using sliding window normalization can ensure that each layer component is in the same distribution. This will further enhance model's generalization ability.”*

3. Only two baselines in the literature are compared. More SOTA methods for multi-task time series classification should be included.

**Reply:** Thank you for your suggestion. We have added two baseline methods. DanHAR and Triple-Res are SOTA methods for HAR tasks. Both use attention to enhance feature extraction capabilities. For the newly added baselines, their training configurations follow the same training configuration of other baselines. The results indicate that MTSDNet still remains optimal. And DanHAR becomes a suboptimal algorithm over GILE on some datasets. We have made modifications to the tables and experiments in main text as follow:

*TABLE IV COMPARISON OF AVERAGE TRAINING TIME FOR ONE EPOCH ACHIEVED BY VARIOUS METHODS ON FIVE DATASETS.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Methods* | *UCIHAR* | *PAMAP2* | *DSADS* | *OPPORTUNITY* | *UniMib* |
| *GRU* | *91.79* | *64.96* | *81.87* | *44.07* | *94.85* |
| *LSTM* | *90.97* | *64.65* | *81.23* | *21.87* | *89.48* |
| *MLP* | *90.52* | *64.61* | *79.47* | *68.50* | *96.29* |
| *Transformer* | *90.57* | *66.20* | *84.29* | *47.47* | *95.77* |
| *DeepConvLSTM* | *92.16* | *69.86* | *85.02* | *54.73* | *94.94* |
| *MTSDNet-A-tsg* | *91.67* | *71.92* | *91.28* | *78.63* | *98.92* |
| *MTSDNet-A-t3s3g3* | *91.70* | *73.34* | *92.16* | *78.14* | *98.85* |
| *MTSDNet-M-tsg* | *92.32* | *72.91* | *91.28* | *78.30* | *98.81* |
| *MTSDNet-M-t3s3g3* | *92.42* | *72.50* | *90.90* | *74.58* | *98.74* |
| *DANN* | *89.29* | *63.3* | *81.30* | *55.49* | *94.75* |
| *GILE* | *92.26* | *72.37* | *87.49* | *58.32* | *97.00* |
| *DanHAR* | *91.60* | *71.73* | *89.46* | *70.12* | *97.26* |
| *Triple-Res* | *91.93* | *71.69* | *86.85* | *73.20* | *96.70* |

**Reviewer #1:**

This paper targets mitigating the impact on the predicting accuracy caused by the inconsistent distribution of training and test data, such as different walking patterns among variant people. The authors propose a new method “Multichannel Time Series Decomposition Network” (MTSDNet) which has good performance in predicting accuracy and stability. The method has the ability to learn universal features to generalize across domains. The original signal, a combination of multiple polynomials and trigonometric functions, is decomposed into low-rank representation. Several experiments are implemented to demonstrate the effcacy of the proposed method. After reading, I have several concerns and questions listed below.

Reply to Reviewer #1:

Thank you for your comments. We have addressed your concerns point-by-point as follows, and revised our manuscript according to your suggestion.

Major Comment

1. As for the structure of the proposed method, I am confused about the statement that the three layers represent Trend Signal, Season Signal and General Signal. Can you explain the difference between General Signal and error items? By the error analysis or parameter estimate?

**Reply:** Thank you very much for your comments. Time-series decomposition used in time-series prediction tasks usually has trend terms, seasonal terms, cyclical terms and residuals. In MTSDNet, time-series decomposition has trend terms, seasonal terms, gerneal terms and errors, respectively. For the convenience of understanding, the terms in two tasks have the following corresponding relationships:

|  |  |
| --- | --- |
| time-series prediction tasks | time-series classification tasks |
| trend | trend |
| seasonal and cyclical | seasonal |
| residuals | general and errors |

The trend items in the two tasks are consistent. There are no complex seasons and periodicity in classification tasks, so seasonal items are used instead of seasonal and periodic items in prediction tasks. Residual in prediction tasks have sporadic signals (general terms) and errors. For example, fall detection relies on sporadic signals. Therefore, MTSDNet uses a general term to capture sporadic signals. The remaining part is error which has no use for any task so it is dropped. We have added the detailed explanation to the main text:

on Page 3:

*“Time series decomposition requires determining which components to construct. In HAR task, time series decomposition in MTSDNet focuses on trend term, seasonal term and general term.*

*The data bias can be captured by the trend item, the periodicity of human activity can be captured by the seasonal item, and the remaining part forms the general term and residual.*

*Here general term is set up to capture sporadic signals. Sporadic signals are often ignored as residual in time series prediction tasks, but in HAR tasks, it is highly correlated with falling.”*

2. Can you explain the weight in the formula ? How do you set in the case study or trained by the model?

**Reply:** Thank you very much for your comments. This formula is used to aggregate the classification results of different components. The weights are consistent with the trainable parameters of the model and are fixed values during model testing. is a weight from a saliency-based attention. Unlike focused attention, saliency-based attention is a type of passive attention. In MTSDNet, the layer attention directly assigns higher weights to specific layers or specific terms, thus achieving fast running speed and small space occupation. For example, trend terms always have higher weights than seasonal terms.

The weight is trained by the model. Taking MTSDNet-tsg as an example, it has three layer attention weight. The specific values of are obtained through an automatic differentiation mechanism during the training process and using softmax to ensure the sum equals 1. When the model predicts, the trained weights are used directly. The result is obtained by simply multiplying the weight and classification logits of each layer. We have added the detailed explanation to the main text:

On Page 4:

*The detailed structure of decomposer is described in subsection III-D. The classifier is implemented using a three-layer MLP. MTSDNet uses a layer attention to aggregate classification results. The layer attention is a saliency based attention. In training stage, data-driven training is used to train the attention weight of each layer of classification results. The attention weight obtained by training reflects the importance of time series decomposition features. Layer attention ensures that the sum of weights is 1 through softmax. When testing, MTSDNet uses the trained attention weight to aggregate multiple classification results.*

3. The value n, m, o in Equation (1) and Equation (2) are the dimensions of variables or the batches of a sample.

**Reply:** We are sorry for your misunderstanding. The value n, m, o in Equation (1) and Equation (2) are the number of layer in trend, seasonal and general term, respectively. Traditional time-series decomposition does not have multiple layers for the same term. For MTSDNet in HAR, only one layer for the same term may not be powerful enough so that the layer from the same term is made stackable.

For example, MTSDNet-A-t3s4g5 means n=3, m=4 and o=5. This also indicates that the model adopts a 3-layer stacking of trend terms, 4-layer stacking of seasonal terms and 5-layer stacking of general terms.

4. Is there any conditions of Equation (3)? What if H > p + 1? Meanwhile, please expand Equation (6).

**Reply:** Sorry for your misunderstanding. The formula has been revised for clearer understanding.

No, Equation (3) does not have any condition. The trend term is always capable of extracting trend features from time-series signal no matter the signal is meaningful or not. Even if the signal is a meaningless noisy one, the trend could still capture the mean value instead of returning an error or raising an exception.

The seasonal term has close to 0 when the periodicity is not significant, which indicates that there is little periodicity.

Here, H is the length of sliding window and p is order of polynomial. H p + 1 is a common thing when H=32 and p=3 while H p + 1 should be paid more attention. The selection of p should not be too large. A large p leads to overfitting and an increase in model complexity.

Equation (6) can be considered as nested functions. We have expanded this nested function for a better understanding.

On Page5:

*Polynomial constraint and trigonometric constraint can be considered as nested functions with time matrix Z as the independent variable. Time matrix Z is used to express the time dimension. Z is defined as follows:*

*Where H is the length of the sliding window and Z is a matrix with size of$1 \* H. The T trend uses the following polynomial constraint:*

*Where T' is a matrix representing polynomial constraints, is the feature generated through trend decomposer and $p$ is the {\color{red}order} of polynomials.*

*F\_T(Z) indicates that the polynomial function acts on each element of Z matrix. F\_T(Z) is equivalent to using Z as the independent variable for p-order polynomial expansion.*

*Polynomial constraints are expressed in matrix form and the decomposition form of the signal is obtained by matrix multiplication with the computed high-dimensional features. This is equivalent to restricting the trend item to a specific structured polynomial function and allowing the model to learn the coefficients of each polynomial term. The S trend item uses the following trigonometric constraint:*

*Where S' is a trigonometric constraint expressed in matrix form, and is the feature generated through seasonal decomposer.*

*F\_S(Z) indicates that the trigonometric function acts on each element of Z matrix. F\_S(Z) is equivalent to using Z as the independent variable for p-order trigonometric expansion.*

5. I cannot catch TABLE 3 in the experiment. How can you determine the number of the parameters of each model? If it stands for parameters, why does there exist a non-integer in the table? Is it the average number of different parts?

**Reply:** Sorry for your confusion. The unit K () is add in TABLE 3, which defines the number of trainable parameters, which is an integer. We are sorry for not adding a unit behind each number.

On Page 8:

*TABLE III Comparison of parameters achieved by various methods on five datasets.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Methods* | *UCIHAR* | *PAMAP2* | *DSADS* | *OPPORTUNITY* | *UniMib* |
| *GRU* | *54.1K* | *65.4K* | *69.7K* | *95.6K* | *51.3K* |
| *LSTM* | *71.9K* | *86.7K* | *92.1K* | *126K* | *68.4K* |
| *MLP* | *165K* | *608K* | *203K* | *945K* | *75.4K* |
| *Transformer* | *437K* | *467K* | *434K* | *453K* | *422K* |
| *DeepConvLSTM* | *86.8K* | *222K* | *222K* | *518K* | *49.4K* |
| *MTSDNet-A-tsg* | *51.4K* | *83.2K* | *75.4K* | *159K* | *64.7K* |
| *MTSDNet-A-t3s3g3* | *154K* | *249K* | *226K* | *479K* | *145K* |
| *DANN* | *216K* | *659K* | *255K* | *996K* | *126K* |
| *GILE* | *225K* | *282K* | *264K* | *415K* | *221K* |
| *DanHAR* | *334K* | *1.9M* | *784K* | *3.8M* | *155K* |
| *Triple-Res* | *193K* | *723K* | *562K* | *1.7M* | *133K* |

From the table t, the unit of DanHAR method on PAMAP2, OPPORTUNITY is M (). The unit of Triple-Res on OPPORTUNITY is also M.

1. How can you determine the number of the parameters of each model?

The model parameters are obtained by a well-designed function from pytorch-lightning library. The function simply traverses each layer of a model and sum up the number of trainable parameters.

1. If it stands for parameters, why does there exist a non-integer in the table?

We are again sorry for misunderstanding. Parameter is actually an integer. The unit has been added behind.

1. Is it the average number of different parts?

Not exactly, the number of trainable parameters is not an average but a constant when model structure is fixed. Trainable parameters are only related to the number of input channels, the number of output classifications and the length of sliding window in the data set. All these hyperparameters are fixed for a model so finally the trainable parameter is a constant.

Minor Comment

1. There exists a typesetting problem, such as the common in Line 39 Page 5/21.

2. Shall you change the notation Sori, as it may cause the mistake as the i-thSor?

3. It would be better to add some details of the model framework or algorithm in the appendix.

**Reply:** Thank you very much for your minor comments. We have checked the typesetting problems, changing notation from to and appending model details in the main text.

1. There exists a typesetting problem, such as the common in Line 39 Page 5/21.

We have checked the typesetting problems:

On Page 6

*For the comparison of experimental results, a variety of traditional methods including LSTM, GRU and three-layer MLP, as well as DeepConvLSTM and Transformer are provided.*

1. Shall you change the notation Sori, as it may cause the mistake as the i-thSor?

We change to . The modification results are as follows:

On Page 5:

1. It would be better to add some details of the model framework or algorithm in the appendix.

In III METHOD, we further elaborate the details of the model.

**Reviewer #2:**

The paper proposes a new method called Multi-channel Time Series Decomposition Network (MTSDNet) for sensor-based activity recognition. The method addresses the problem of out-of-domain generalization in cross-person behavior recognition by decomposing the original signal into multiple components and classifying them layer by layer. The proposed method is evaluated on multiple public datasets and compared with state-of-the-art methods, showing improved prediction accuracy and stability. However, some concerns should be addressed by it can be published.

Reply to Reviewer #2:

Thank you for your comments. We have addressed your concerns point-by-point as follows, and revised our manuscript according to your suggestion.

1. The motivation behind the design of your model is essential. Could you theoretically explain why your model achieves higher accuracy compared to others?

**Reply:** Thank you very much for your comments. In III METHOD, the design motivation of the model is elaborated. Simply, MTSDNet takes advantage of time series decomposition to decouple features and eliminate domain differences.

By decoupling features, domain differences can be limited to specific features within a specific layer. Each layer retains its own features from domains and keeps itself independent from other layers.

For example, when the domain difference only comes from wearing type, only the layers from trend term will recognize the difference, while those from seasonal term and general term are unaware. Even if the classification of layers from trend term is incorrect due to domain differences, layers from seasonal and general term are still able to classify in a correct way because layers are independent from each other. Therefore, time series decomposition manages to enhance the domain generalization ability in HAR tasks.

We also add this detailed explanation to the main text:

*In this section, we will introduce the structural design of the MTSDNet model, explain why time series decomposition is introduced and how time series decomposition can enhance domain generalization ability.*

*For model design, introducing prior knowledge of specific tasks can lead to significant performance improvements. Such as the convolution used by CNN in the field of images introduces prior knowledge that the closer the distance between pixels, the stronger the correlation. LSTM and GRU force data to flow in chronological order to capture temporal characteristics. PINN in the field of physics encodes physical equations into optimization objectives to constrain feasible solutions to satisfy physical laws.*

*sensor-based human activity recognition can be regarded as a time series classification task with domain differences or distribution differences. The differences in HAR are regular. By analyzing the possible differences in human activities, we can get the following errors.*

*Regarding the analysis of the sensor data used for cross-person HAR, it is found that the following factors are the main reasons for the distribution differences between different volunteers:*

*Data bias: arising from the wearing differences of the sensors in the same position and the variations in the three-axis direction caused by the sensor production, this bias is manifested in the original signal as static or background deviation. Taking the accelerometer as an example, the bias appears as variations in data direction and amplitude, contingent upon a variety of factors, including sensor placement, rotation, motion-induced displacement and so on.*

*Volunteer activity differences: mainly derived from the signal differences reflected on sensors under the amplitude, frequency and composite of dynamic actions between different volunteers, as well as signal differences of complex actions on the sensors. This difference is manifested in the original signal as features such as the amplitude, frequency and periodicity of seasonal signals.*

*Secondary noise: mainly arising from the relative movement of sensors to the body caused by human motion when the sensors and the body are not tightly attached. The information collected by the sensors contains the movement of the sensors relative to the body in addition to the human motion, which is manifested as data bias changing over time in other original signal.*

*For the above, data bias is rarely singled out and addressed specifically. The work[16] explains the data bias problem caused by the sensor hardware, but does not consider the data bias problem caused by the fixed way of the sensor on human body.*

*Domain differences in HAR are different from those in image fields. Complex domain differences can be regarded as the superposition of the above-mentioned errors. A simple idea is to deal with these differences separately. MTSDNet uses time series decomposition to decouple errors and features. Time series decomposition can decouple the domain differences and simplify the problem.*

*Time series decomposition are often used for time series prediction tasks such as predicting geographic and financial data. In these tasks, different components are discarded or retained because of different task concerns. The task that focus on long-term trends needs to eliminate the effects of seasonal terms and residuals, vice versa. Time series decomposition can eliminate the interaction between different components.*

*Time series decomposition requires determining which components to construct. In HAR task, time series decomposition in MTSDNet focuses on trend term, seasonal term and general term.*

*The data bias can be captured by the trend item, the periodicity of human activity can be captured by the seasonal item, and the remaining part forms the general term and residual.*

*Here general term is set up to capture sporadic signals. Sporadic signals are often ignored as residual in time series prediction tasks, but in HAR tasks, it is highly correlated with falling.*

*By using time series decomposition, Specific domain differences will be limited to a certain component. For example, the data bias caused by wearing is only generated in the trend item. When there is only data bias, temporal decomposition can ensure that the seasonal and general terms are not disturbed. Even if the classification of trend term is incorrect due to domain differences, seasonal and general terms can be correctly classified. Therefore, time series decomposition can enhance the domain generalization ability in HAR tasks.*

*What's more, time series decomposition is helpful to explicit feature alignment. Explicit feature alignment explicitly makes the feature distributions of multiple source domains as close as possible. By learning a decomposition, each layer of components is located in the same feature space. Using sliding window normalization can ensure that each layer component is in the same distribution. This will further enhance model's generalization ability.*

2. Is your model sequential? Obtaining trend signals through one decomposer and then obtaining seasonal signals through another decomposer increases the complexity of model training. In such cases, how robust and stable are the prediction results?

**Reply:** Thank you very much for your comments.

1. Is your model sequential?

Yes, MTSDNet is sequential. The model name is self-explanatory. For example, MTSDNet-tsg indicates extracting features in the order of trend term, seasonal term, and general term, respectively.

1. In such cases, how robust and stable are the prediction results?

The stability can be inferred from the variance of 10 independent training results, while the robustness can be judged by focusing on different volunteer divisions. The competitive accuracy performance of MTSDNet on the OPPORTUNITY dataset where strong noise exists can further prove that the method is robust enough.

In Fig 4 and Fig 5, the length of the ribbon shows the stability of the model. MTSDNet has the shortest ribbon length and higher accuracy on all volunteers.

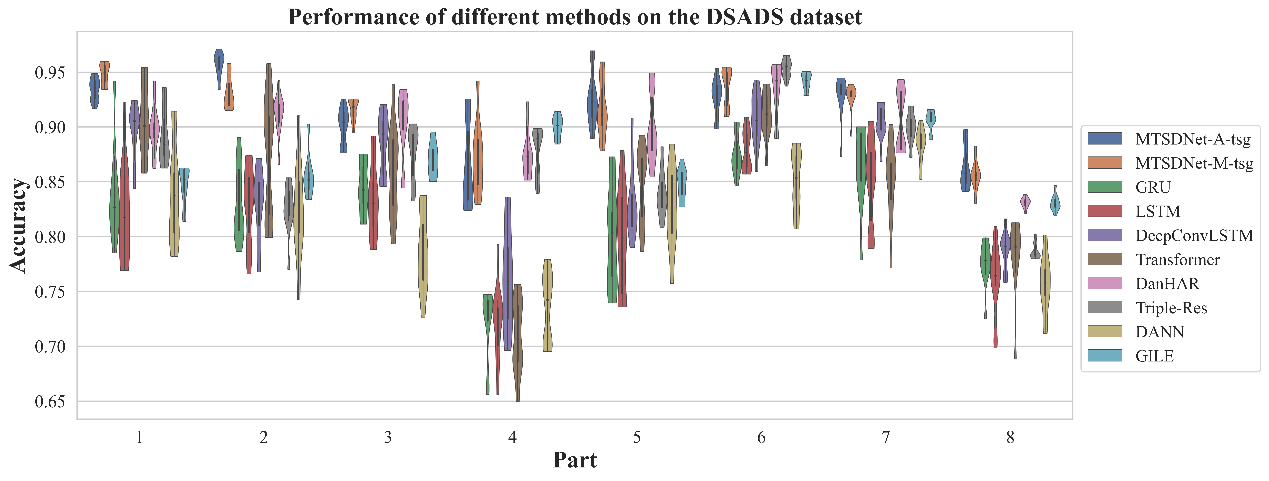


Fig. 4. Violin plot of accuracy distribution on the DSADS dataset. MTSDNet shown in blue and orange achieves higher accuracy than other baselines on

almost all Parts.

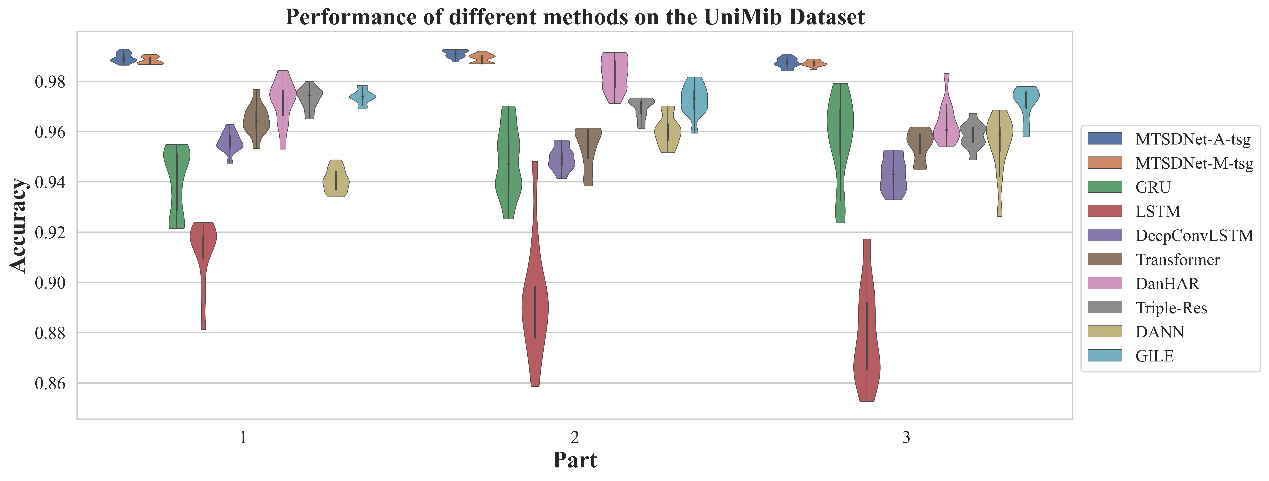


Fig. 5. Violin plot of accuracy distribution of UniMib dataset. MTSDNet shown in blue and orange achieves higher accuracy than other baselines on 3 parts

and also has higher stability.

Meanwhile, specific results on the OPPORTUNITY dataset is presented.

TABLE VIII THE ACCURACY COMPARISON OF DIFFERENT METHODS ON THE OPPORTUNITYDATASET FROM PART1TO PART4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| METHOD | Part1 | Part2 | Part3 | Part4 |
| GRU | 48.29(5.81) | 57.70(8.64) | 32.63(5.82) | 37.68(6.94) |
| LSTM | 24.69(5.49) | 25.14(4.78) | 17.83(4.38) | 19.81(3.79) |
| MLP | 75.14(3.33) | 70.49(3.97) | 58.80(7.05) | 69.57(4.10) |
| Transformer | 50.94(6.88) | 56.2(13.05) | 35.52(11.01) | 47.24(12.98) |
| DeepConvLSTM | 56.45(26.11) | 61.48(5.21) | 43.83(4.34) | 57.17(20.55) |
| MTSDNet-A-tsg | 82.67(0.33) | 78.97(1.05) | 71.89(4.88) | 81.00(0.94) |
| MTSDNet-A-t3s3g3 | 81.24(6.18) | 79.50(1.09) | 69.96(4.80) | 81.87(0.47) |
| MTSDNet-M-tsg | 82.37(0.30) | 79.08(0.55) | 71.77(4.83) | 79.98(0.49) |
| MTSDNet-M-t3s3g3 | 82.37(0.29) | 78.74(0.67) | 56.64(24.31) | 80.59(0.41) |
| DANN | 57.25(8.38) | 64.91(4.74) | 47.47(10.82) | 52.34(8.63) |
| GILE | 60.09(10.20) | 67.66(7.76) | 49.41(6.32) | 56.11(7.61) |
| DanHAR | 71.87(7.74) | 73.63(5.97) | 63.5(6.19) | 71.48(3.82) |
| Triple-RES | 79.24(0.58) | 73.31(3.02) | 70.9(2.48) | 69.34(5.23) |

The OPPORTUNITY dataset is collected in a real environment. The dataset contains a large amount of noise and includes only 4 volunteers.

The data in TABLE VIII represents the average accuracy and accuracy variance obtained from 10 training sessions. Taking PART1 as an example, MTSDNet-A-tsg has the highest average accuracy (82.67) and almost the smallest variance (0.33). It can be seen that when facing data with noise, the average accuracy of MTSDNet is significantly better than other baseline methods.

The above results prove that MTSDNet has strong robustness and stability.

3. How does this model compare with existing methods in terms of time efficiency and complexity?

**Reply:** Thanks for your comments. In order to better demonstrate the advantages of MTSDNet, we add the average training time of one epoch in the main text.

In average training time, MTSDNet-tsg is between LSTM and GILE. The difference in training time is more significant on the PAMAP2 dataset. Overall, MTSDNet can balance model size and computation efficiency while ensuring the highest average accuracy.

We also add a table to the main text:

On Page 8:

*TABLE IV COMPARISON OF AVERAGE TRAINING TIME FOR ONE EPOCH ACHIEVED BY VARIOUS METHODS ON FIVE DATASETS.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Methods* | *UCIHAR* | *PAMAP2* | *DSADS* | *OPPORTUNITY* | *UniMib* |
| *GRU* | *53.89* | *194.06* | *64.78* | *70.78* | *46.39* |
| *LSTM* | *52.27* | *200.37* | *64.16* | *70.26* | *48.19* |
| *MLP* | *51.18* | *148.06* | *62.09* | *71.93* | *43.84* |
| *Transformer* | *67.75* | *540.87* | *98.48* | *90.5* | *69.36* |
| *DeepConvLSTM* | *62.58* | *569.1* | *116.81* | *133.9* | *59.61* |
| *MTSDNet-A-tsg* | *56.98* | *332.42* | *103.41* | *93.88* | *49.82* |
| *MTSDNet-A-t3s3g3* | *71.8* | *618.49* | *158.12* | *125.14* | *62.27* |
| *DANN* | *50.76* | *180.9* | *74.3* | *73.26* | *45.77* |
| *GILE* | *61.21* | *385.34* | *115.01* | *100.9* | *57.26* |
| *DanHAR* | *57.59* | *697.33* | *94.64* | *178.15* | *47.77* |
| *Triple-Res* | *58.24* | *697.16* | *117.57* | *196.13* | *50.34* |

*Table III shows the parameters of multiple models on five datasets and Table IV shows average training time for one epoch. The parameters of MTSDNet-tsg is approximately equal to LSTM and it can exclude the improvement of accuracy caused by the increase in model size. This also indicates that the improvement in MTSDNet accuracy does not come from the increase in parameters. MTSDNet-A-tsg can achieve better classification accuracy with relatively reasonable model parameters.*

*In Table IV, the average training time of MTSDNet-A-tsg is second only to MLP, GRU, LSTM, and DANN. MTSDNet-A-t3s3g3 with higher stacking are closer in training time to the slowest method. DANN can be approximately equivalent to MLP.*

*In the comparison of traditional methods, DanHAR and Triple-Res, which design new attention module for HAR tasks, have significant advantages. However, the shortcomings of these two models lie in their size and time requirements. DanHAR has the largest size among all models and can reach about 24 times the size of MTSDNet-A-tsg on OPPORTUNITY.*

*Overall, MTSDNet can balance model size and operational efficiency while ensuring the highest average accuracy.*

4. Your algorithm is a classification algorithm. How is prediction implemented?

**Reply:** Sorry for misunderstanding. MTSDNet is a classification algorithm and is not intended to make predictions. The reason for this issue may be the improper use of prediction in the original text. We have made corrections to the use of prediction in the text.

In abstract:

*“Extensive evaluation on DSADS, OPPORTUNITY, PAMAP2, UCIHAR and UniMib public datasets shows the advantages in classifying accuracy and stability of our method compared with other competing strategies, including the state-of-the-art ones.”*

*“This paper is motivated by solving the problem of decreased classification accuracy caused by differences in human activity, such as the differences in walking patterns between young and elderly people.”*

On Page 9:

*“And the difference of classification accuracy and stability between the additive model and the multiplicative model is small.”*