MACHINE LEARNING LABORATORY SERIES

Combined Machine Learning Logbook Labs 1 2 3 and 4

Classical models feedforward networks and recurrent sequence models

Chathuranga Bandara Polkotuwa.

Date and lab session: 11/09/2026

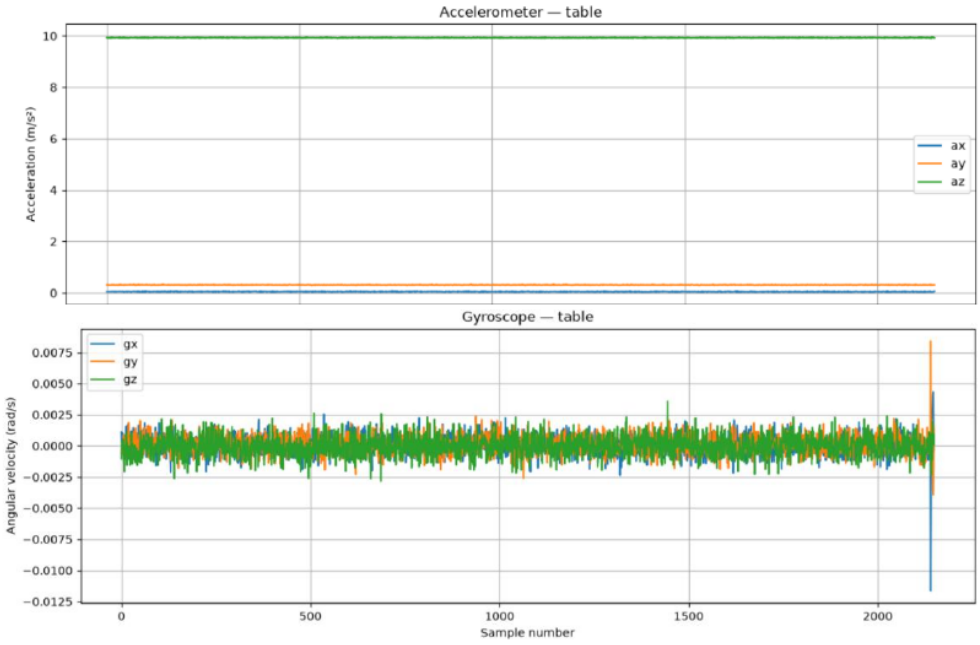
What I did: I imported my accelerometer and gyroscope CSV files into pandas and checked the column names and data structure. I created DataFrames with ax, ay, az for the accelerometer and gx, gy, gz for the gyroscope. I plotted the signals for the table, upright, and circle recordings and compared the differences. I then created a function to read and combine accelerometer and gyroscope files, saved the processed recordings as new CSV files, and created another function to combine recordings from the same class with a class label.

Choices and settings: I used only the six required sensor features: ax, ay, az, gx, gy, gz. I used clear filenames for each recording and used a continuous index when combining several files. I also saved processed CSV files without the pandas index because the index is not needed as a sensor feature.

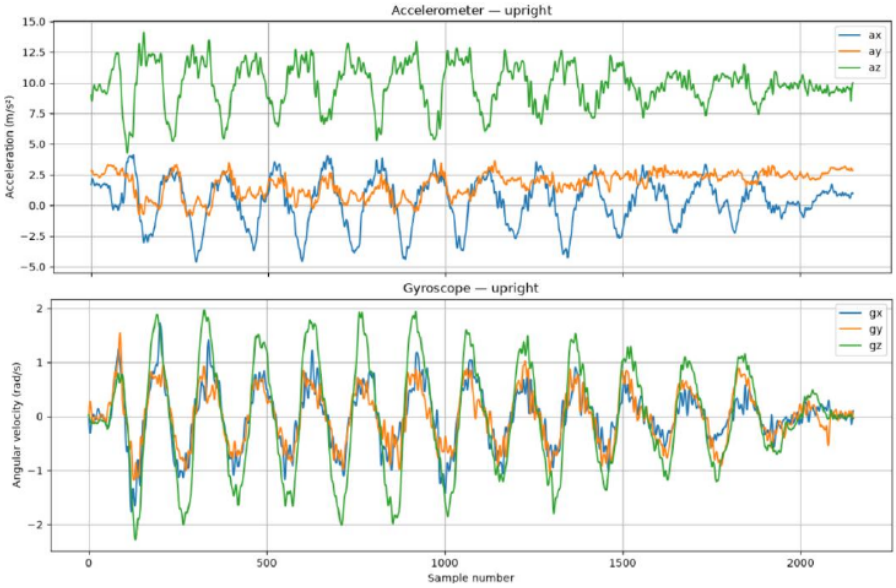
Code/commands used:

- I used pd.read_csv() to import the CSV files, pandas functions to select and combine columns, pd.concat() to combine DataFrames, and Matplotlib to plot the sensor signals. I also used to_csv() to save the processed recordings
- Function for read the accelerometer data, read the gyroscope data, combine the sensor data into one DataFrame, return a DataFrame containing the six sensor features: ax, ay, az, gx, gy, gz and lable for the class. Also this function save the result according to given format.

Result: Table

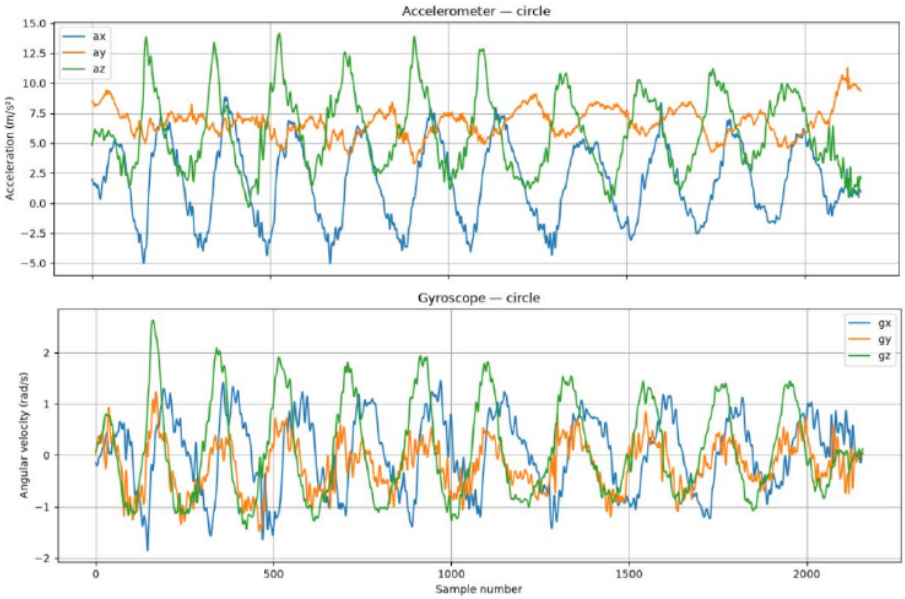


Plots of upright movement.



Plots of circle movement.

I



successfully created processed DataFrames containing the six sensor features. The stationary table and upright recordings showed relatively stable accelerometer signals, while the circle recording showed more variation. The gyroscope values were mostly stable when the phone was stationary and changed more during circular movement. The final combined class DataFrames contained ax, ay, az, gx, gy, gz, class.

Comment/issues: I had to check the actual CSV column names before using the generated code because the file structure can differ between phones. I also checked that the accelerometer and gyroscope recordings matched correctly before combining them. I learned that the accelerometer also measures the effect of gravity, while the gyroscope mainly measures rotational movement.

Practical Lab 2 Logbook

Classical Machine Learning with Movement Sensor Data | Grade 5

A total of 48 movement recordings were organized and used to evaluate KNN and Decision Tree classifiers. Model selection was based on recording-aware cross-validation. The selected model achieved 91.12% accuracy on the untouched same-person test set and was then evaluated on recordings from another person.

Dataset organization and split

The dataset contained 12 recordings for each class (walking, running, jumping and own movement), giving 95,837 samples in total. Accelerometer and gyroscope rows were aligned to the nearest timestamp within 0.02 s. Each row contained ax, ay, az, gx, gy and gz also its class label and recording_id.

Using a fixed random seed, 10 recordings per class were assigned to training (40 recordings; 79,795 samples), while 2 per class were reserved for the final test set (8 recordings; 16,042 samples). No recording_id values were shared between the sets. Related samples were therefore kept together and leakage was prevented.

Training data exploration and quality

There were no missing sensor values. The distributions of all six sensor axes were compared across the four movement classes. Acceleration differed most clearly for jumping and running, while the own movement usually had a narrower range. The gyroscope axes also showed wider rotational variation during running than during the lower-intensity own movement.

The boxplots show , no single feature completely separated every class. Instead, useful differences were spread across the acceleration and angular-velocity axes. This supported the use of all six measurements as model inputs.

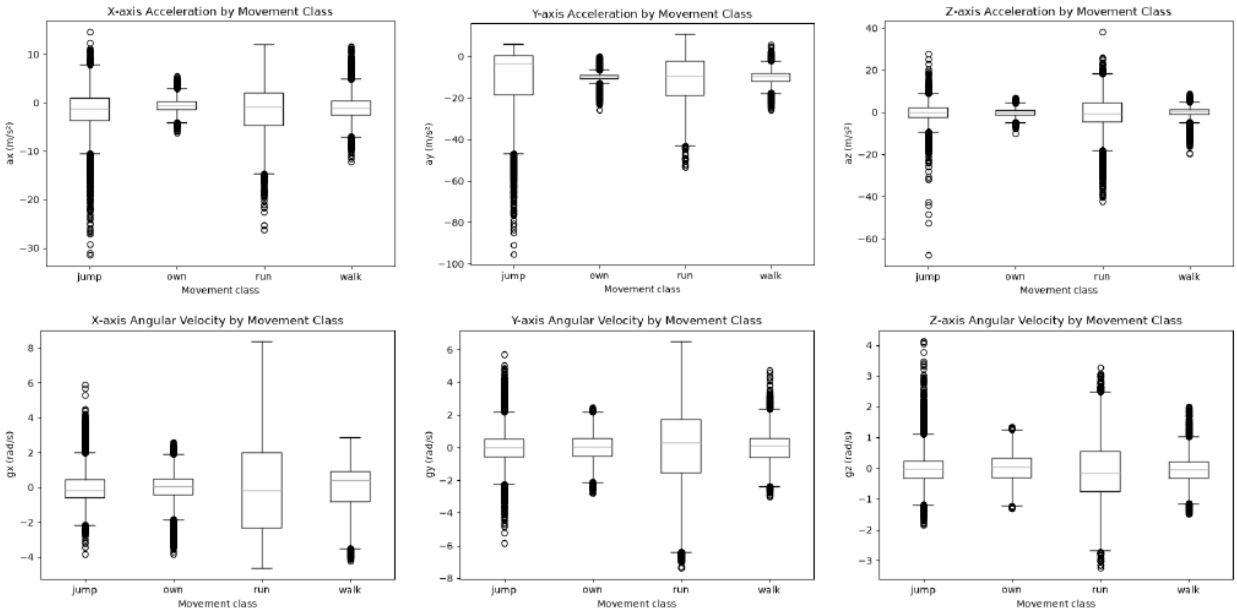


Figure 1 Training distributions for ax, ay, az, gx, gy and gz by movement class

Model development and Grade 3 results

The feature matrix used only ax, ay, az, gx, gy and gz; the class was the target and recording_id supplied the groups. Five-fold StratifiedGroupKFold created 32 training and 8 validation recordings per fold with no recording overlap.

Grade 3 KNN and Decision Tree

For KNN, odd k values from 1 to 15 were tested. For the Decision Tree, max_depth values of 3, 5, 7, 10 and None were compared using gini and entropy. Tree depth had much more influence than criterion. The best Decision Tree used entropy with no depth limit and achieved 89.44% mean cross-validation accuracy. Its 100% training accuracy indicated overfitting, even though it was the strongest tree configuration.

Feature scaling

Scaling was expected to affect KNN because its distances would otherwise be dominated by the larger acceleration ranges. StandardScaler was placed inside a pipeline and fitted only on the training portion of each fold. Decision Trees were left unscaled because scaling changes thresholds but not the ordering of values.

Model	Best setting	Mean CV	SD
KNN unscaled	k=5	91.44%	2.44%
KNN + StandardScaler	k=7	93.42%	1.85%
Decision Tree	entropy, depth=None	89.44%	2.28%

Grade 3 final model selection

Scaled KNN with k=7 was selected as the Grade 3 final model because its 93.42% mean cross-validation accuracy was higher than both unscaled KNN (91.44%) and the Decision Tree (89.44%). Scaling also reduced the variation between folds. After selection, the untouched test set was evaluated once and produced 91.04% accuracy. The confusion matrix shows that running was classified most reliably, while walking was most often confused with the own movement.

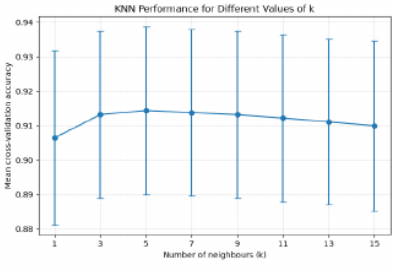


Figure 2 KNN accuracy by k

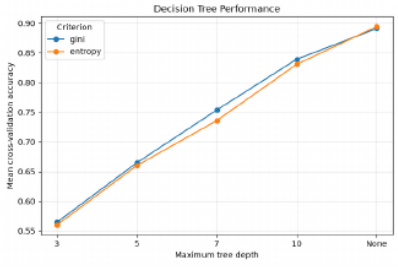


Figure 3 Decision Tree validation accuracy

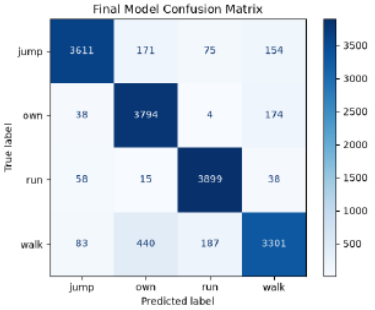


Figure 4 Grade 3 final-model confusion matrix

Grade 4 optimization and stability

The extended scaled-KNN grid tested k=3, 5, 7, 9 and 11; uniform versus distance weights; and Euclidean versus Manhattan distance. These ranges focused on the strongest Grade 3 region while testing whether closer neighbours should contribute more strongly. The extended tree grid tested gini/entropy, depths 10, 15, 20 and None, min_samples_split 2, 5 and 10, and min_samples_leaf 1, 2 and 5 to examine regularization.

Model	Best setting	Mean CV	SD
Extended scaled KNN	Euclidean, k=11, distance	93.62%	1.87%
Extended Decision Tree	entropy, depth=20, split=2, leaf=1	89.55%	2.24%

Variation between folds

Extended KNN scored 94.18%, 94.06%, 93.91%, 90.15% and 95.82% across the five folds (range 5.67 points). The tree scored 90.37%, 89.89%, 90.39%, 85.26% and 91.82% (range 6.57 points). Fold 4 was hardest and Fold 5 easiest for both models, indicating differences in the assigned recordings rather than instability unique to one algorithm.



Figure 5 Performance of the two extended models across folds

How clear was the choice

The best KNN setting had only a very small advantage. Euclidean distance weighting with k=11 scored 93.62%, while k=7 scored 93.61%, and several configurations were within 0.2 percentage points. However, the choice between KNN and Decision Tree was clear. Extended KNN outperformed the extended tree in every fold by about four points on average and was also slightly more stable. The extended scaled KNN was therefore selected. The Grade 4 search improved test accuracy only from 91.04% to 91.12%, which matched the small cross-validation differences between the leading KNN settings.

Final evaluation and deeper error analysis

The selected StandardScaler - KNN pipeline (Euclidean distance, k=11, distance weighting) achieved 91.12% on the final test set. This was 2.50 percentage points below the 93.62% cross-validation mean. The result was slightly lower than expected, but it remained close enough to support the selected model.

Class	Precision	Recall	F1
Jump	95.87%	89.78%	92.73%
Own	85.71%	94.66%	89.96%
Run	93.83%	97.16%	95.47%
Walk	89.72%	82.87%	86.16%

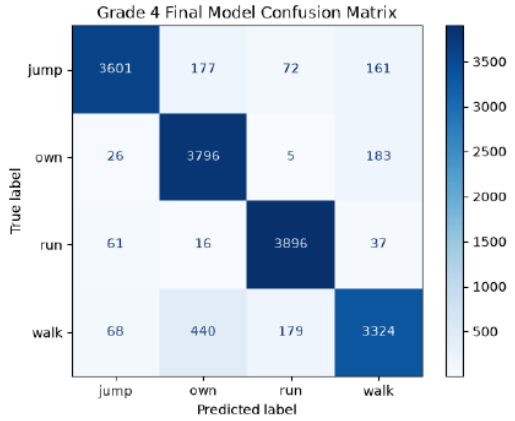


Figure 6 Grade 4 final confusion matrix

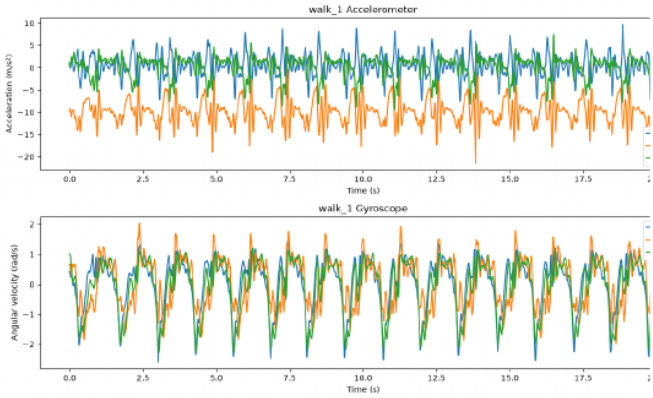


Figure 7 Difficult walk_1 test recording

Error patterns and improvement

Running was classified best, with 97.16% recall, while walking was the most difficult, with 82.87% recall. The most common error was walking being classified as the own movement, which happened for 440 samples. Most errors were found in `walk_1`, which had 75.32% accuracy, and `jump_1`, which had 85.89% accuracy. The walking recording still showed a clear repeated pattern, while the jumping recording contained jumps with different strengths. The model classified each sensor row separately, so some individual rows looked similar to another movement even when the full recording showed a clear pattern.

The model could be improved by dividing each recording into short overlapping time windows. Features such as the mean, standard deviation, minimum, maximum, range, signal magnitude and main frequency could then be calculated for each window. This would help the model recognize repeated walking patterns and jump impacts. Collecting more recordings with different movement speeds and strengths could also help the model work better on new data.

Grade 5 - Generalization to another person using an external dataset

The final StandardScaler - KNN model was selected using only the original training data. No recordings from the other person were used for model selection, parameter tuning or fitting. The unchanged model was applied to 48 external recordings containing 95,441 aligned samples, with 12 recordings available for each movement class.

Evaluation set	Recordings	Samples	Accuracy
Own untouched test set	8	16,042	91.12%
Other person's external data	48	95,441	37.94%

Between-person accuracy was 53.18 percentage points lower than within-person accuracy. This large reduction shows that strong performance on unseen recordings from the same person did not transfer directly to a new person.

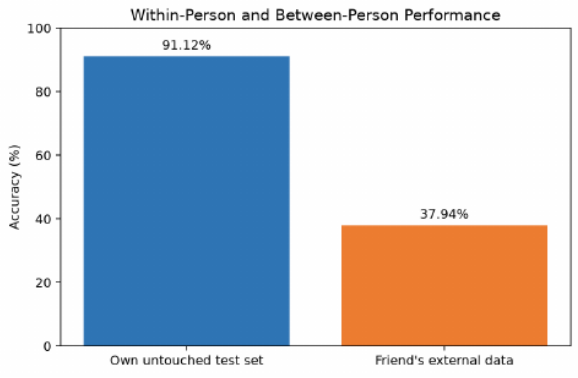


Figure 8 Within-person and between-person accuracy

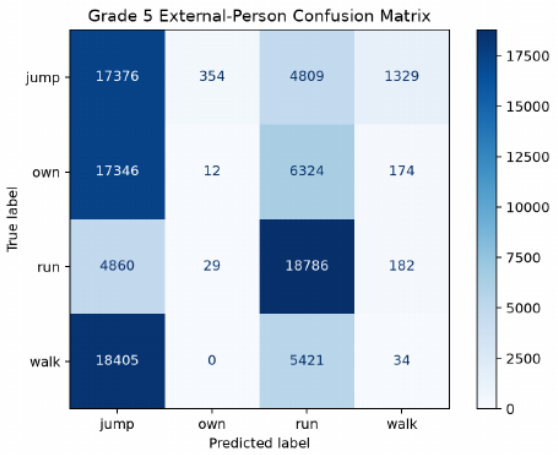


Figure 9 External-person confusion matrix

Comparison between-person performance (two different dataset)

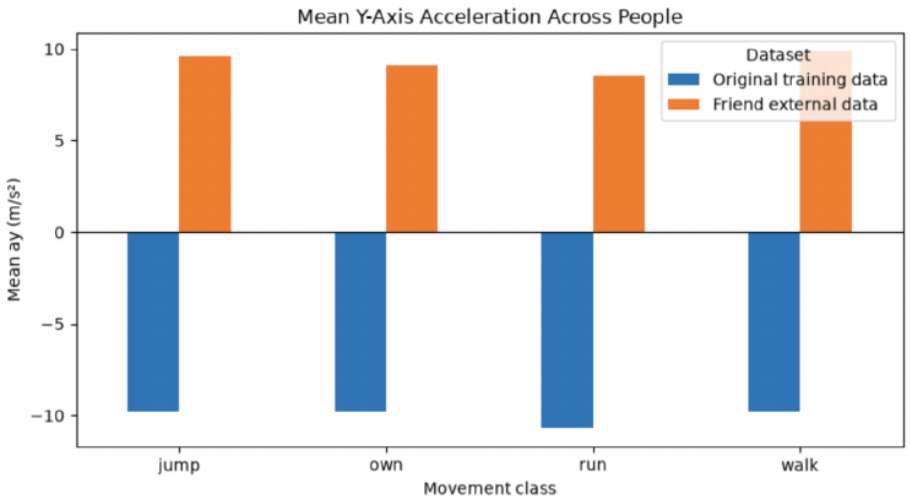


Figure10 comparison , Mean y axis between two dataset

Running and jumping transferred better than the two lower-intensity classes, with recalls of 78.74% and 72.80%. In Other hand, recall was only 0.05% for the own movement and 0.14% for walking. Most samples from these classes were predicted as jumping or running. The confusion matrix therefore shows a systematic distribution shift rather than a small loss of accuracy across every class.

How well the model works for another person

One clear difference was how the phone was positioned. The average ay value was around −10 m/s^2 in the original training data, but around +9 m/s^2 in the friend's data. This suggests that the phone was facing a different direction. Differences were also seen on the other sensor axes, especially during running and walking. These changes may have been caused by phone placement, movement style or how the phone was handled. Since KNN compares the distance between sensor values, these differences made the friend's movements look different from the original training data.

The model achieved 91.12% accuracy on test data from the same person, but only 37.94% on the friend's data. This shows that the model worked well when the test data was collected in a similar way, but it did not work as well for a new person. The model could be improved by collecting training data from more people, keeping the phone in a similar position and using features that are less affected by phone orientation. Future testing could also keep one person completely outside the training data.

Use of generative AI: ChatGPT was used to help explain the code, solve errors and organize the report. The notebook was run, and the results were checked before they were included in the report.

Practical Lab 3 Logbook

Four movements - walk, run, jump and own movement - were classified from the six sensor features ax, ay, az, gx, gy and gz. The same 48 recordings from Lab 2 were used. There were 12 recordings per class and 95,837 aligned sensor samples in total. The final Lab 2 test recordings were kept untouched until the neural-network configuration had been selected.

Data part	Recordings	Samples	Purpose
Development training	32	63,849	Model-weight fitting
Validation	8	15,946	Configuration comparison
Final test	8	16,042	One final evaluation

The data were split by recording_id rather than by individual rows. This was important because neighbouring samples from one recording are strongly related. If rows from the same recording appeared in both training and validation data, the validation score would be unrealistically optimistic. Eight development recordings and two validation recordings were contributed by each class.

Environment and data check

Accelerometer and gyroscope files were loaded from the sensor data folder. The timestamps were aligned to the nearest value within 0.02 seconds, while the label and recording_id were retained. All six sensor columns were confirmed, and no missing sensor values were found.

First network with 10 neurons

A Sequential network was built with six inputs, two Dense hidden layers containing 10 ReLU neurons each, and a four-neuron softmax output. Nonlinear relationships were learned through ReLU. One probability for each movement class was produced by softmax. Adam and sparse categorical cross-entropy were used because the targets were integer class labels.

The first model contained 224 trainable parameters and was trained for 50 epochs with a batch size of 256. A validation accuracy of 87.56% was reached on unscaled data. The training and validation curves remained close, so no strong overfitting was observed. However, limited capacity in the small network was suggested by the result.

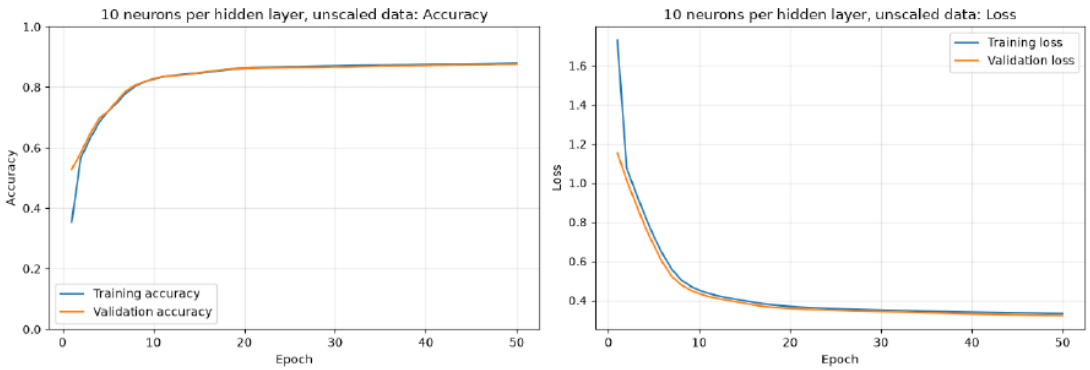


Figure 1 Training history for the 10 neuron unscaled model

Larger network with 50 neurons

The same input, output, activation, loss and optimizer choices were retained, but both hidden layers were changed to 50 neurons. The number of trainable parameters was increased from 224 to 3,104. Faster learning was observed, and 93.95% validation accuracy was reached with a validation loss of 0.1746. The validation curve stayed close to the training curve, so the additional capacity was shown to be useful without obvious overfitting.

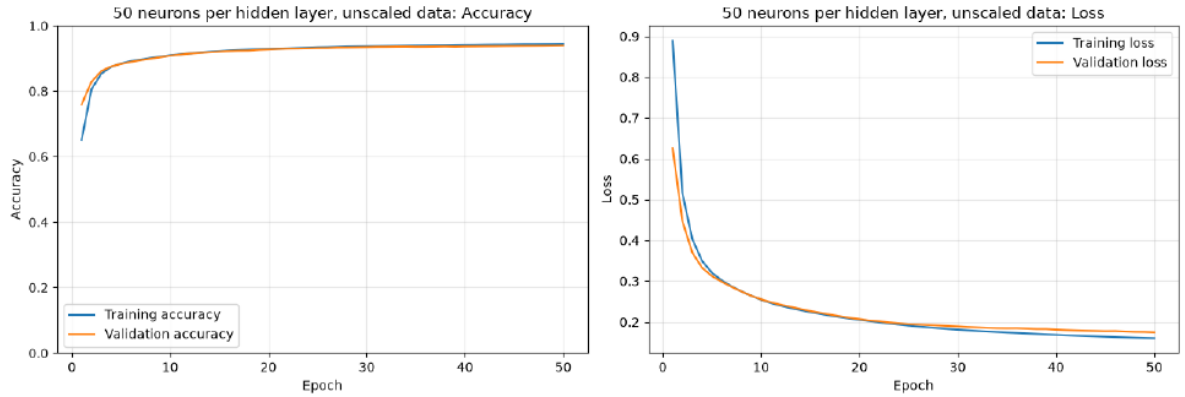


Figure 2 Training history for the 50 neuron unscaled model

Feature scaling

Different numerical ranges were present across the acceleration and gyroscope axes. Large-scale features were expected to influence gradient updates more strongly and potentially make learning slower or less stable. StandardScaler was therefore fitted only on the development training features. The same fitted scaler was applied to the validation data. The scaled training features had means close to zero and standard deviations close to one.

After scaling, the best validation accuracy of the 10-neuron model was increased from 87.56% to 88.79%. A best validation accuracy of 94.76% was reached by the scaled 50-neuron model at epoch 47, with a final validation loss of 0.1531. A modest improvement was therefore produced by scaling for both model sizes.

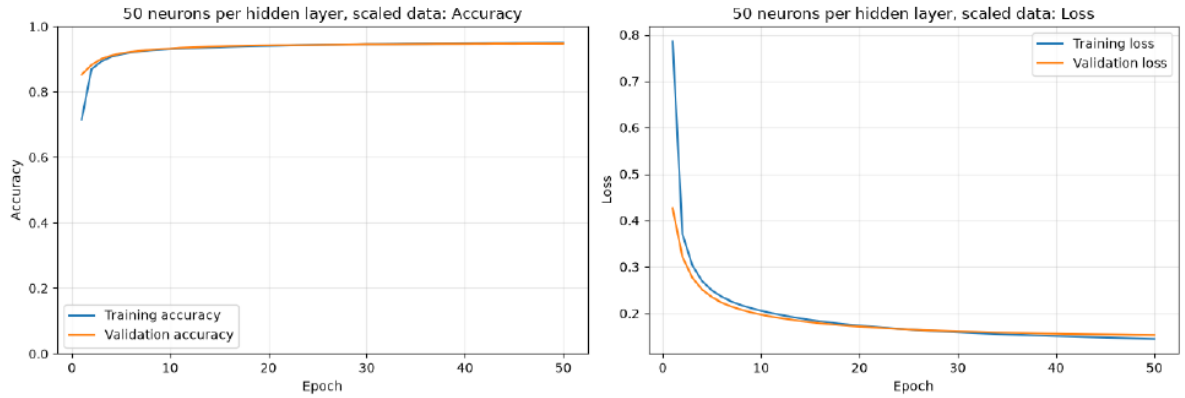


Figure 3 Training history for the selected 50 neuron scaled model

Results and model selection

Model	Parameters	Final val accuracy	Final val loss	Best val accuracy	Best epoch
10 neurons unscaled	224	87.56%	0.3232	87.56%	50
50 neurons unscaled	3,104	93.95%	0.1746	93.95%	50
10 neurons scaled	224	88.75%	0.3113	88.79%	48
50 neurons scaled	3,104	94.73%	0.1531	94.76%	47

The scaled 50-neuron model was selected because the highest validation accuracy, the lowest validation loss and stable learning curves were obtained. This decision was made before the final test set was used. To use the strongest validation point, the model was recreated with the same random seed and was trained for 47 epochs.

Final test evaluation

The untouched test features were transformed with the scaler fitted on the development training data. The scaler was not refitted on the test set. A test accuracy of 90.09% and a test loss of 0.3416 were achieved by the final model. Its macro-average F1-score was 0.8995. Greater difficulty in the unseen recordings was indicated by the 4.67 percentage point drop from best validation accuracy.

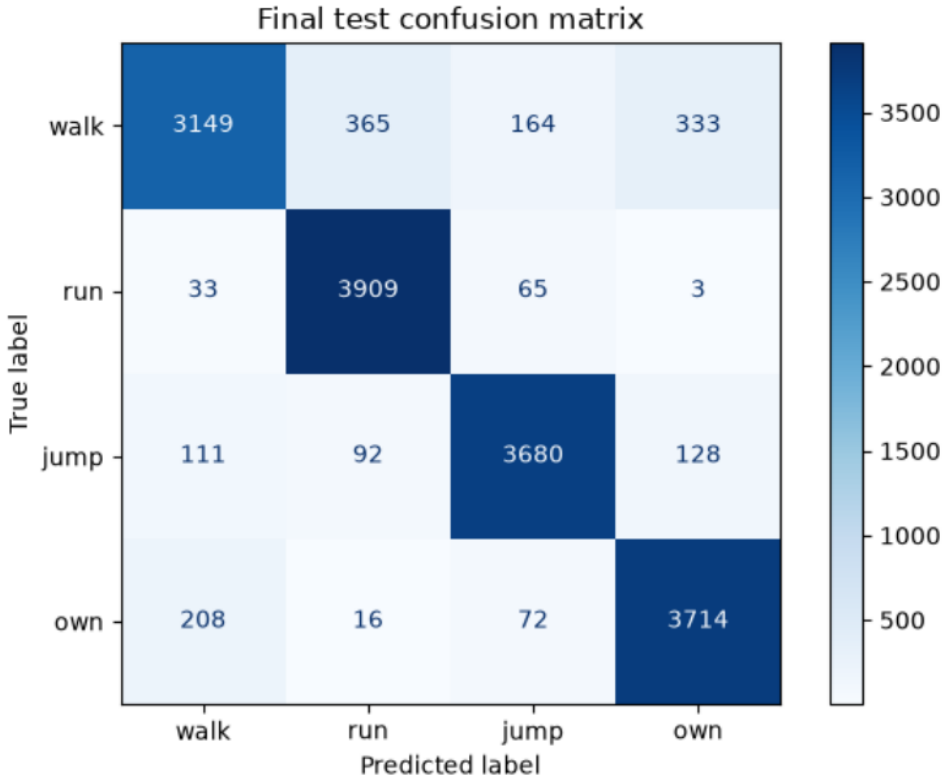


Figure 4 Confusion matrix for the untouched final test recordings

Running was the easiest class, with 97.48% recall and 3,909 of 4,010 samples classified correctly. Walking was the hardest, with 78.51% recall. Walking was most often predicted as running (365 samples) or the own movement (333 samples). Jump and own movement had recalls of 91.75% and 92.62%.

The test features and true labels were used by Keras evaluate() to return overall loss and accuracy. Four probabilities per sample were returned by predict(). Argmax was used to choose the predicted class, and those class indices were compared with the true labels to create the confusion matrix. Predictions were needed because evaluate() does not return the individual predicted classes.

Code and commands used

Following the feedback on the Lab 2 logbook, the key commands were recorded below. These are short command references rather than the full source code, which remains in the submitted Jupyter notebook.

Command or method	How it was used
pd.read_csv and pd.merge_asof	Both sensors were loaded and their nearest timestamps were aligned
recording_id grouping	Complete recordings were kept in one split
StandardScaler fit and transform	The scaler was fitted on training features and applied to validation and test features
keras.Sequential and Dense	The 6 to hidden layers to 4 network was created
model.fit	Training was performed for 50 epochs and validation history was recorded
model.evaluate and model.predict	Metrics were calculated and sample-level probabilities were produced
confusion_matrix	True and predicted movement classes were compared
model.save and joblib.dump	The final model and fitted scaler were saved for Lab 4

Important code snippets

Only the main implementation patterns are shown below. The complete executable code is retained in the Jupyter notebook.

Recording aware data selection

```
train_data = full_data[full_data["recording_id"].isin(train_ids)]
test_data = full_data[full_data["recording_id"].isin(test_ids)]
```

Feedforward model definition

```
model = keras.Sequential([
    keras.Input(shape=(6,)),
    layers.Dense(50, activation="relu"),
    layers.Dense(50, activation="relu"),
    layers.Dense(4, activation="softmax")
])
model.compile(optimizer="adam",
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])
```

Training with separate validation data

```
history = model.fit(
    X_train_scaled, y_train,
    validation_data=(X_validation_scaled, y_validation),
    epochs=50, batch_size=256
)
```

Scaling without data leakage

```
X_train_scaled = scaler.fit_transform(X_train_unscaled)
X_validation_scaled = scaler.transform(X_validation_unscaled)
X_test_scaled = scaler.transform(X_test_unscaled)
```

Final evaluation and class predictions

```
test_loss, test_accuracy = final_model.evaluate(X_test_scaled, y_test)
probabilities = final_model.predict(X_test_scaled)
y_test_predicted = np.argmax(probabilities, axis=1)
```

What was learned and what could be improved

The clearest result was that model capacity had a larger effect than scaling. The biggest accuracy increase was produced by moving from 10 to 50 neurons per hidden layer, while an additional smaller improvement was produced by scaling. It was also shown that a high validation score does not guarantee the same performance on new recordings. This difference was made visible by keeping recording groups separate.

Walking was identified as the main weakness, especially when individual samples looked similar to running or own movement. Each sensor row was treated independently by the feedforward model, so movement order and rhythm were not used. In Lab 4, better performance is expected when sequences are used because walking and jumping contain patterns across time that are not visible in one row. The final Keras model, fitted scaler and feature/class mapping were saved so the same preprocessing setup can be continued.

Use of generative AI

ChatGPT was used step by step to help structure the notebook, explain TensorFlow and Keras choices, troubleshoot the missing TensorFlow kernel, and interpret the learning curves and confusion matrix. Every notebook cell was run, and the shapes, splits and outputs were checked.

Lab 4 - Recurrent Neural Networks with GRU

Aim and outcome

A four-class movement classifier was developed from accelerometer and gyroscope sequences. A gated recurrent unit network was selected by validation performance, and a final controlled-test accuracy of 99.19% was obtained. Strong controlled performance was not transferred to the friend recording, for which an approximate accuracy of 25.78% was obtained.

Experimental scope

- Walking, running, jumping and forward or backward movement were classified from six sensor features.
- Window length, GRU width, recurrent depth, dropout and learning rate were compared without use of the test recordings.
- Continuous prediction, comparison with the Lab 3 feedforward model and person-generalization testing were included.

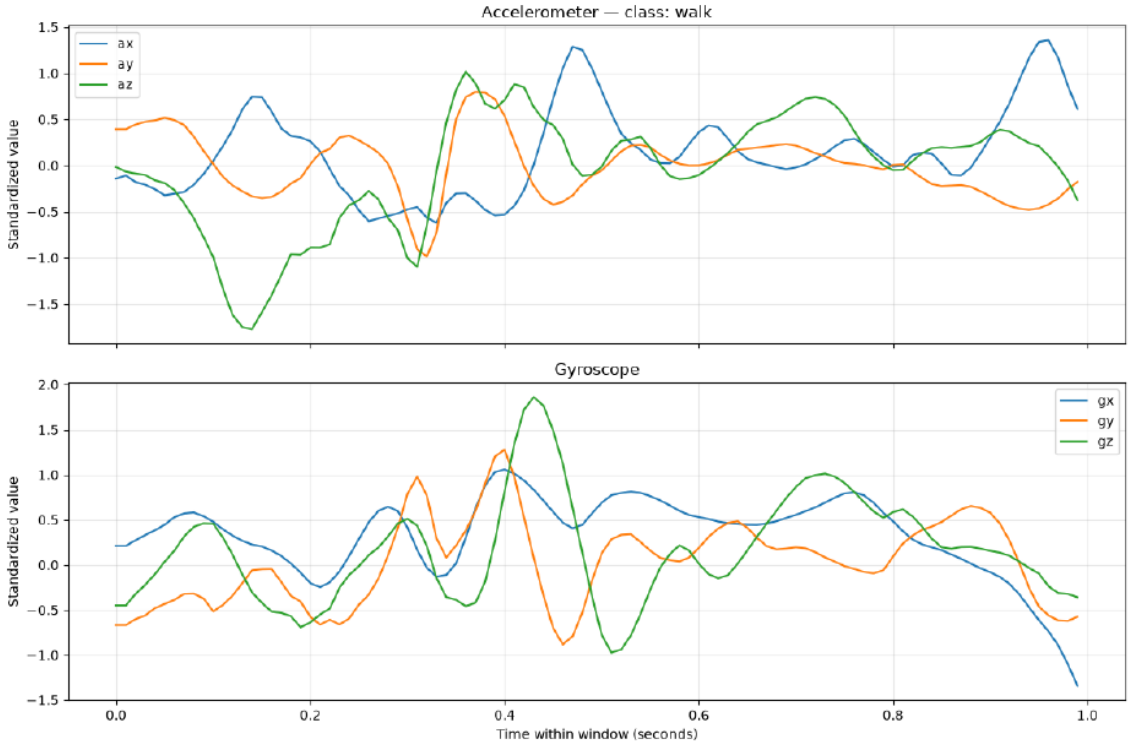


Figure 1 One standardized one-second walking window. Three accelerometer and three gyroscope channels were retained.

Sequence modelling was used because motion class information is carried by the order and repetition of sensor measurements. In contrast, a feedforward model is supplied with a fixed feature vector and does not preserve the same temporal state between timesteps.

Data preparation and baseline model

Forty-eight recordings were used, with 12 recordings provided for each class. Recording-level separation was applied so that windows from one recording could not appear in more than one subset.

Stage	Implementation	Reason
Split	32 train, 8 validation, 8 test recordings	Leakage between recordings was prevented
Scaling	StandardScaler fitted on training samples only	Validation, test and external data remained unseen
Windowing	100 timesteps x 6 features, step 1	One second and 99% overlap were represented
Boundary rule	Windows were created inside each recording	Movement and recording boundaries were not crossed

Baseline architecture

A single GRU layer with 32 units was followed by a four-unit softmax output layer. Sparse categorical cross-entropy was used because one integer class label was assigned to each window. Adam was used for optimization. Early stopping was controlled by validation loss, and the best weights were restored.

Input	GRU	Output	Parameters
100 x 6	32 units	4 class probabilities	3,972

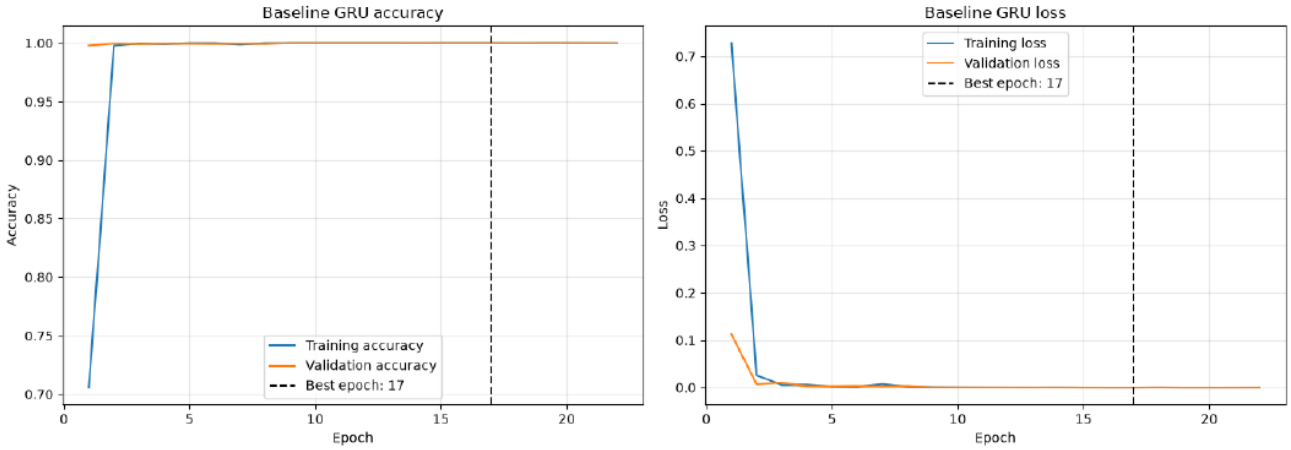


Figure 2 Baseline accuracy and loss. Rapid convergence was observed and the best validation result was reached at epoch 17.

The baseline was completed in 22 epochs. A validation accuracy of 100% and a validation loss of 0.000113 were recorded. Because adjacent windows overlapped by 99%, window-level observations were highly correlated and the near-perfect validation accuracy was interpreted together with recording-level separation and external-person testing.

Validation based model selection

Only training and validation recordings were used during model selection. The test recordings were kept untouched until the complete configuration had been fixed.

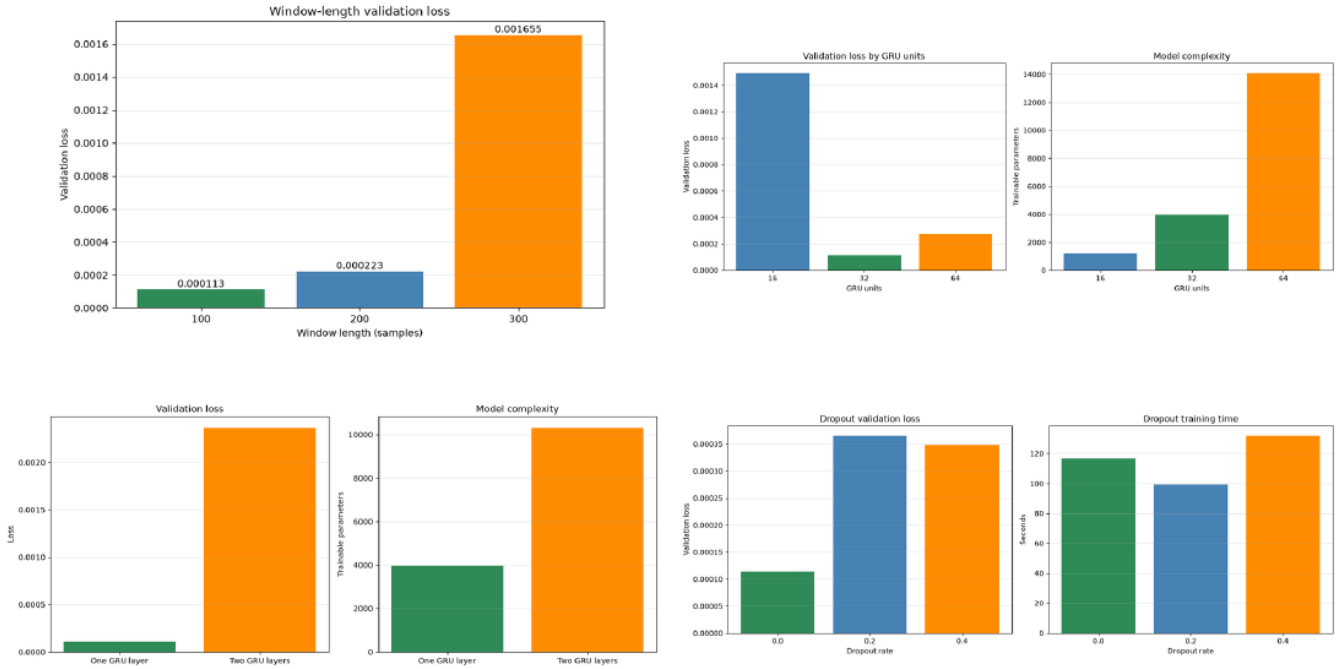


Figure 3 Validation losses obtained during window, width, depth and dropout comparisons. Green indicates the selected setting in each panel.

Experiment	Candidates	Selected	Validation evidence
Window length	100, 200, 300 samples	100	Lowest loss 0.000113
GRU units	16, 32, 64	32	Lowest loss 0.000113 with 3,972 parameters
GRU layers	1, 2	1	Two layers increased loss to 0.002365
Dropout	0.0, 0.2, 0.4	0.0	Dropout did not improve validation loss

Selection interpretation

The one-second window was retained because the lowest validation loss was produced while the largest number of training windows was preserved. Sixteen GRU units produced slightly lower capacity and a validation loss of 0.001491. Sixty-four units reduced training loss further, but validation loss increased to 0.000274 and the parameter count rose to 14,084. The 32-unit model therefore provided the strongest validation result with moderate complexity.

No benefit was obtained from a second GRU layer. Its parameter count increased from 3,972 to 10,308, while validation loss increased by more than one order of magnitude. Increased dropout also raised training loss without improving validation loss. The simpler one-layer, zero-dropout architecture was therefore retained.

Grade 4 tuning and final controlled test

Learning rates of 0.0001, 0.001 and 0.003 were compared on the fixed split. A rate of 0.001 was selected because the lowest validation loss, 0.000113, was obtained. Although 0.003 required less training time, a slightly higher validation loss of 0.000124 was produced.

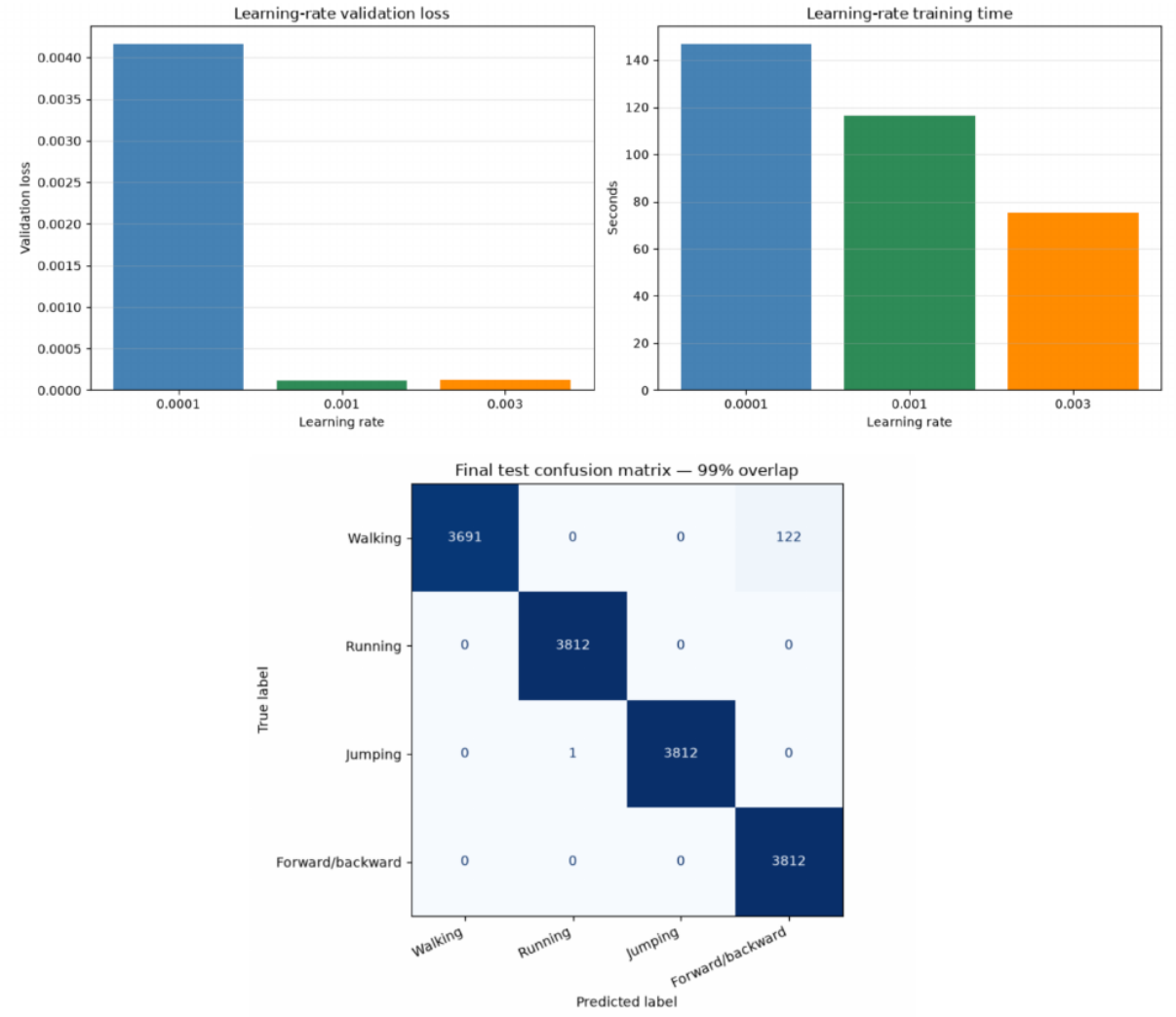


Figure 4 Learning-rate tuning and the final controlled-test confusion matrix.

Final setting	Value	Final result	Value
Window and step	100 and 1	Validation accuracy	100.00%
Units and layers	32 and 1	Validation loss	0.000113
Dropout and rate	0.0 and 0.001	Test accuracy	99.19%
Batch size	256	Test loss	0.043053

Test Result

Of 15,250 overlapping test windows, 15,127 were classified correctly. All running and forward or backward windows were classified correctly. One jumping window was classified as running, and 122 walking windows were classified as forward or backward. Because these errors were counted across strongly overlapping windows, consecutive mistakes may have originated from short shared signal regions.

The Lab 3 feedforward result of 90.09% was exceeded by 9.10 percentage points. The improvement was attributed to the temporal order retained by the GRU. Additional Grade 4 searches required approximately 570 seconds of recorded GPU training time, but the best model remained the compact baseline configuration. Added architectural cost was therefore not justified by validation performance.

Continuous movement recording

The frozen model and training-fitted scaler were applied to a 100-second group recording sampled at approximately 100.22 Hz. A total of 9,924 one-second windows were produced with a one-sample step, giving a prediction interval of approximately 0.01 seconds.

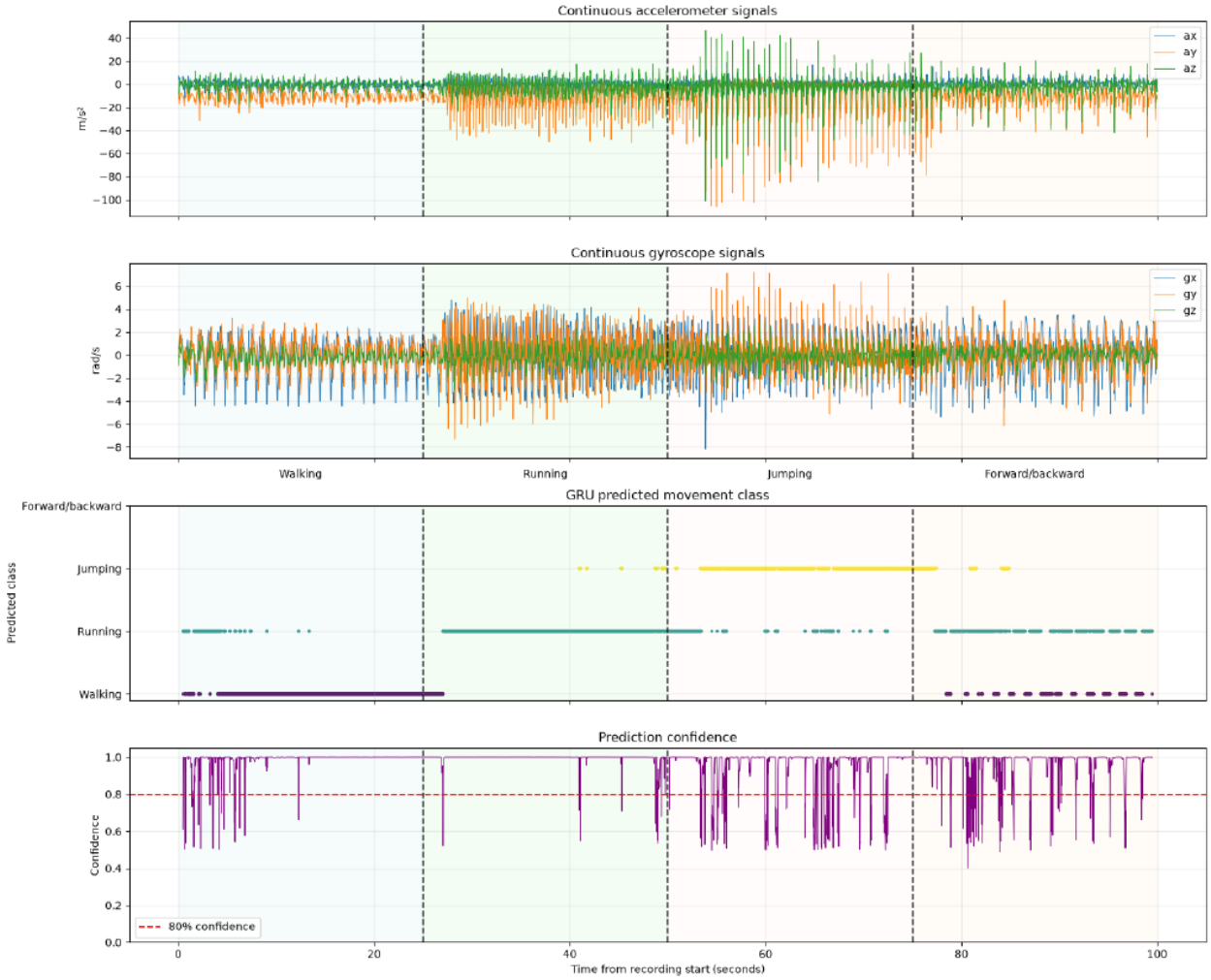


Figure 5 Continuous accelerometer and gyroscope signals, predicted movement class and prediction confidence. Dashed lines indicate approximate activity boundaries.

Approximate section	Windows	Section agreement	Mean confidence
Walking	2,457	86.20%	0.990
Running	2,505	89.06%	0.995
Jumping	2,506	78.41%	0.951
Forward & backward	2,456	0.00%	0.968

Transition and confidence analysis

Stable walking, running and jumping regions were generally identified. Confidence reductions and rapid label changes were observed near activity transitions. No forward or backward output was produced in the final quarter; 59.24% of those windows were classified as running, 27.73% as walking and 13.03% as jumping. This result indicates that the fourth continuous activity was not represented in the same way as the controlled training movement, despite high softmax confidence.

Grade 5 - Person generalization and conclusions

The saved model and scaler were applied without retraining to a friend's 98.16-second recording. The same six features, 100-sample window and 99% overlap were used. Equal-duration sections were used as approximate labels because exact event annotations were unavailable.

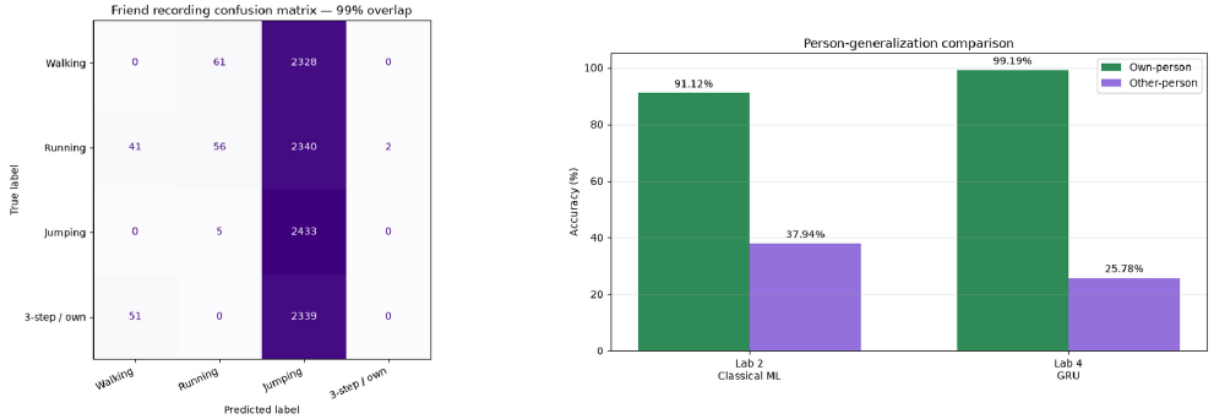


Figure 6 Approximate friend-recording confusion matrix and comparison of own-person and other-person results in Labs 2 and 4.

Grade 5 findings

An approximate friend-recording accuracy of 25.78% was obtained. Of 9,656 windows, 9,440 were predicted as jumping. The controlled-test accuracy therefore decreased by 73.42 percentage points when another person and recording setup were introduced. In Lab 2, a decrease from 91.12% to 37.94% had been observed, corresponding to 53.18 percentage points. The two laboratory arrangements were not identical, so the size of the decreases was compared descriptively rather than as a controlled causal result.

Limitations

- The friend sections were estimated from time rather than supplied as exact labels.
- Three-position stepping was performed in the final friend section, whereas forward or backward movement had been learned as the fourth model class.
- Phone orientation, placement, movement intensity and individual technique may have shifted the six sensor distributions.
- The 99% overlap produced correlated windows, so window-level accuracy did not represent independent decisions.

Conclusion

A compact one-layer GRU was selected successfully and excellent performance was obtained on held-out recordings from the controlled data collection. Temporal modelling improved the controlled result relative to the Lab 3 feedforward model. However, person-independent performance was not achieved. Generalization should be improved through multi-person training data, consistent class definitions, orientation normalization or augmentation, and evaluation on independently labelled recordings from unseen participants.

Overall concluding section

The best result under the controlled recording conditions was obtained with the GRU model. Better movement recognition was achieved when one-second sequences were used. However, much lower results were obtained on the continuous recording and on another person. Therefore, the GRU would be selected for the controlled task, but more data and further testing would be needed before wider use.

What was learned about the sensor data

Walking, running, jumping and forward/backward movement were classified. Six sensor features were used: ax, ay and az from the accelerometer, and gx, gy and gz from the gyroscope. In Lab 1, the effect of gravity was observed in the acceleration readings, even when the phone was still. Changes in rotation were measured by the gyroscope. Different information was therefore provided by the two sensors.
A total of 48 recordings was used, with 12 recordings per class and 95,837 aligned sensor rows. The sensor streams were matched by the nearest timestamp within 0.02 seconds. Stronger signal changes were generally observed during running and jumping. Similar individual readings were sometimes found in walking and forward/backward movement. The importance of phone position and orientation was also observed.

Validation results

Model	Accuracy	How it was measured
KNN	93.62%	Mean of 5 grouped cross-validation folds
Decision Tree	89.55%	Mean of 5 grouped cross-validation folds
Feedforward network	94.76%	Best accuracy on the validation set
GRU	100.00%	Accuracy on the validation set

Different validation methods were used. The cross-validation means should therefore be compared with the single validation-set results with care.

Final test results

Model	Test accuracy	Prediction input
KNN	91.12%	One sensor row
Decision Tree	Not reported	One sensor row
Feedforward network	90.09%	One sensor row
GRU	99.19%	100 sensor rows in one window

The KNN result was rounded to 91% on the workshop board. The exact notebook result is shown above. No final test accuracy was reported for the Decision Tree. Different prediction inputs were used for the GRU and the row-based models, so the scores should not be treated as fully equivalent measurements.

Settings tested and models selected

Model settings were compared with training and validation data. The main choices are summarized below. “Selected” is used because the choice was based on the settings tested in these labs.

Feedforward neural network settings

Setting	Values tested	Selected value
Neurons in each hidden layer	10 and 50	50
Input scaling	Unscaled and standardized	Standardized
Training epoch	Validation results over 50 epochs	47

Two hidden layers were used. When the number of neurons per layer was increased from 10 to 50, unscaled validation accuracy was improved from 87.56% to 93.95%. After standardization, the best result of the larger network was increased to 94.76%. A final network with six inputs, two layers of 50 ReLU neurons and four outputs was selected. A total of 3,104 parameters was used.

GRU settings

Setting	Values tested	Selected value
Window length	100, 200 and 300 samples	100 samples
GRU units	16, 32 and 64	32
GRU layers	1 and 2	1
Dropout	0.0, 0.2 and 0.4	0.0
Learning rate	0.0001, 0.001 and 0.003	0.001

The lowest validation loss was obtained with the compact GRU. One GRU layer with 32 units and four output classes was selected. A total of 3,972 parameters and a batch size of 256 were used. No improvement in validation loss was obtained by adding a second GRU layer or dropout. The selected validation loss was approximately 0.000113.

Classical model settings

For KNN, 11 neighbours, distance weighting and Euclidean distance were selected after the extended search. Standardization was included in the pipeline. For the Decision Tree, entropy, a maximum depth of 20, a minimum split size of 2 and a minimum leaf size of 1 were selected in the extended search. Better grouped cross-validation results were obtained with KNN, so KNN was selected as the final classical model.
Only a small improvement was obtained from the extra KNN tuning. A larger search was therefore not automatically followed by a large improvement in the final result. The same point was seen in Lab 4, where the simpler GRU configuration was retained.

What was learned from comparing the models

From individual rows to time windows

In Lab 2, a class was assigned by KNN using nearby training samples. With a Decision Tree, a class was assigned through rules based on feature thresholds. Scaling was needed for KNN because distances could otherwise be dominated by features with larger numerical values. Less effect from scaling was expected for a Decision Tree because the order of feature values is preserved after scaling.
In Lab 3, a nonlinear relationship between the six inputs and the movement labels was learned by the feedforward network. Each row was still treated separately. A test accuracy of 90.09% was obtained, compared with 91.12% for KNN. The lowest class recall was obtained for walking: 78.51% with the feedforward model and 82.87% with KNN. Similar readings from different movements could not always be separated by using one row.

In Lab 4, 100 consecutive rows were placed in each GRU window. About one second of movement was represented at a sampling rate of about 100 Hz. Changes over time could then be used during classification. A total of 15,127 out of 15,250 test windows was classified correctly. Walking recall was increased to 96.80%. All running and forward/backward windows were classified correctly, and only one jumping window was classified incorrectly.

Was temporal information useful

An improvement of 9.10 percentage points over the feedforward network and about 8.08 points over KNN was obtained with the GRU. These results were taken as evidence that temporal information was useful under the controlled conditions. Repeated changes within a movement could be used instead of relying on one sensor row. However, the full improvement cannot be attributed only to time order because different input representations and models were used. A comparison using the same windows, with and without time order, would be needed to check this more directly.

How data leakage was reduced

Whole recordings were kept separate during splitting. In Lab 2, 40 recordings were used for grouped cross-validation and eight were held out for testing. In Labs 3 and 4, 32 recordings were used for training, eight for validation and eight for testing. Scalers were fitted only on training data. Validation data were used for model choices. Test data were intended to be used after these choices had been made.

Test results were reported at both the Grade 3 and Grade 4 stages of Lab 2. The same test data were therefore inspected more than once during the series. In a future study, a separate final test set should be kept until all model comparisons have been completed.

GRU windows were created within each recording, using a step of one sample. An overlap of 99% was produced. Direct sharing of recordings between splits was avoided, but almost the same samples were included in neighbouring test windows. The 15,250 test windows should therefore not be counted as 15,250 independent examples. Only eight test recordings were used.

Results from the continuous recording

A duration of 100.0 seconds was measured in the saved notebook for the recording described as approximately 90 seconds in the seminar preparation. The saved GRU and training scaler were used without further fitting. A total of 9,924 overlapping windows was produced. The activity sections were estimated from time, so the results below should be treated as approximate agreement rather than exact accuracy.

Estimated section	Windows	Agreement	Mean confidence
Walking	2,457	86.20%	0.990
Running	2,505	89.06%	0.995
Jumping	2,506	78.41%	0.951
Forward/backward	2,456	0.00%	0.968

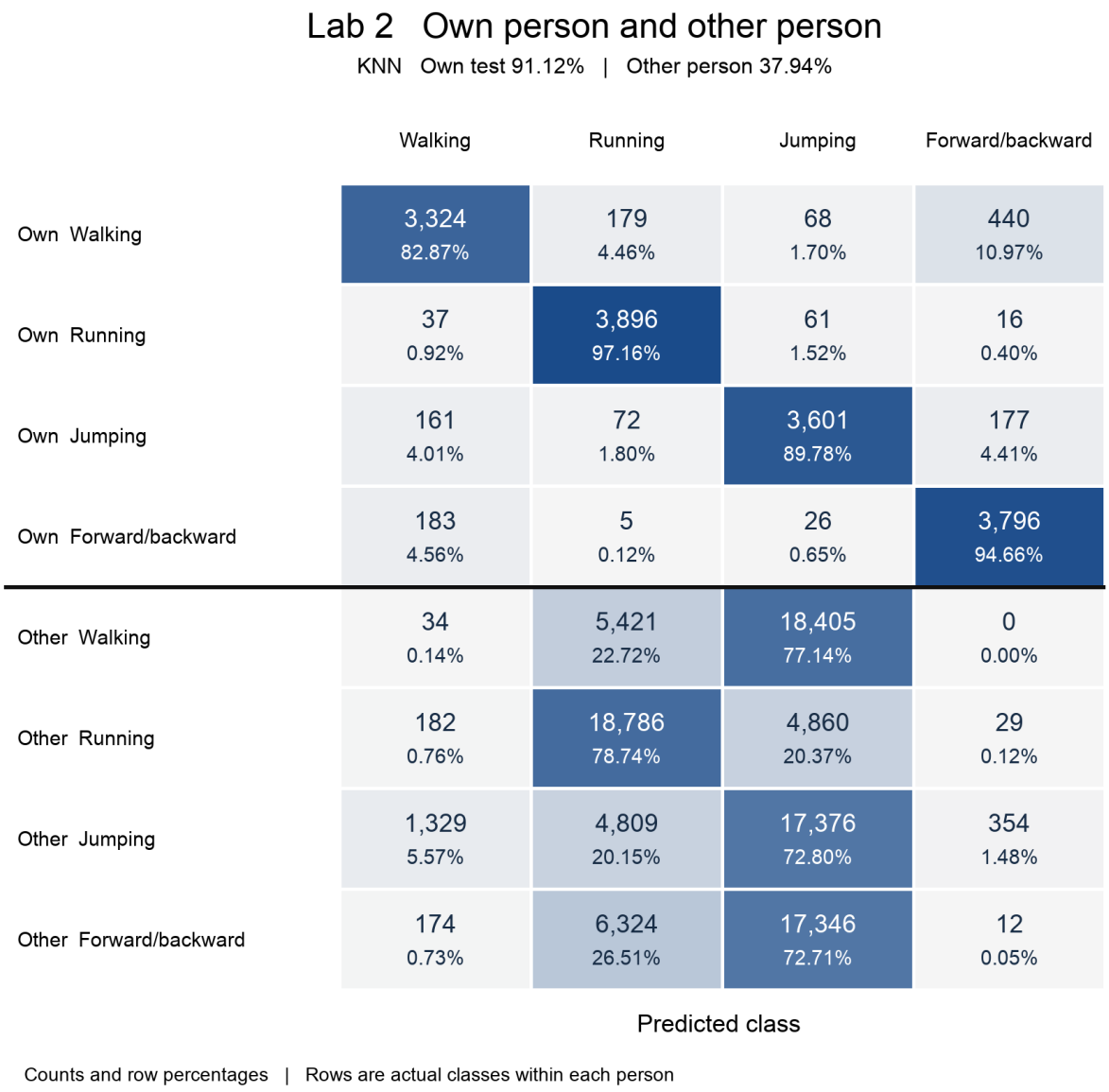
An overall agreement of 63.62% was obtained with the estimated section labels. Walking, running and jumping were recognized during much of their expected sections. Rapid label changes and lower confidence were observed near several activity changes. Parts of two movements could be included in one window near a transition, so a single correct label could be difficult to assign.

The final forward/backward section was not recognized. Of its windows, 59.24% were labelled as running, 27.73% as walking and 13.03% as jumping. A high mean confidence of 0.968 was still obtained. High prediction confidence should therefore not be accepted as proof that a movement was classified correctly.
Much better results were obtained on the controlled test recordings than on the continuous recording. Differences in movement speed, phone position, orientation or movement execution may have been involved. However, no single cause was established. More realistic continuous data and accurate transition labels would be needed to investigate this failure.

How the person comparisons were arranged

Eight actual-class rows and four prediction columns are shown in each of the following matrices. The four classes from the own-person test are shown first, followed by the four classes from the other person. Counts and percentages within each row are included. “Own person” refers to the participant, while forward/backward refers to the fourth movement class.
The models and scalers were used without refitting on the other person. In Lab 2, 48 separate recordings with 95,441 rows were evaluated. In Lab 4, one continuous recording of about 98.16 seconds with 9,656 windows was evaluated. Different data and labels were used in these two external tests. The results should therefore be compared with care.

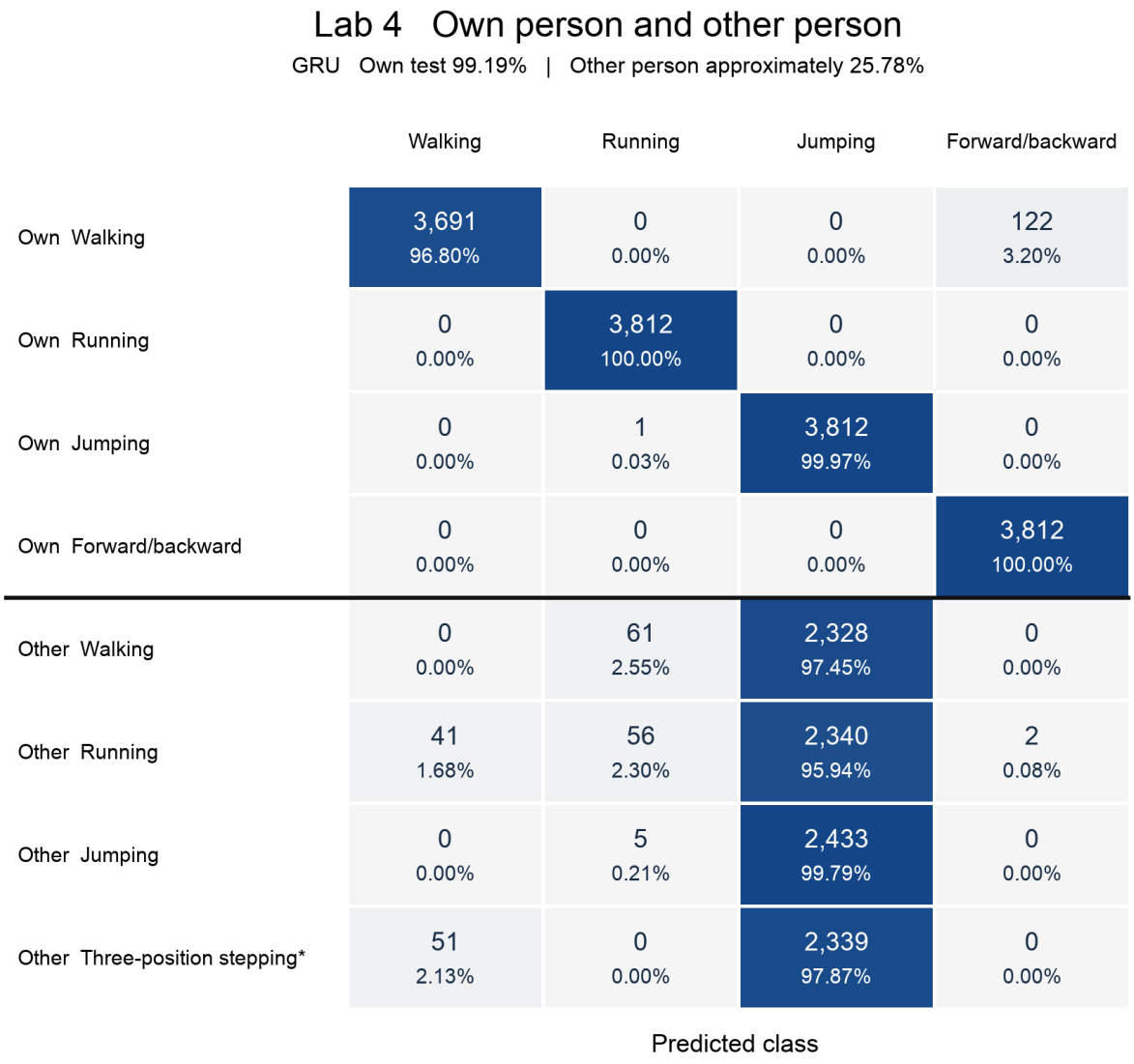
Lab 2 results on another person



A test accuracy of 91.12% was obtained on the own-person recordings, but only 37.94% was obtained on the other person. A decrease of 53.18 percentage points was measured. Walking recall was reduced to 0.14%, and recall for the fourth movement was reduced to 0.05%. Better results were obtained for running and jumping, with recalls of 78.74% and 72.80%. Most low-intensity external movements were labelled as jumping or running.

A change in mean a_y from about -10 m/s² to about $+9$ m/s² was found in the feature comparison. A difference in phone orientation or placement was suggested by this change. This difference was not removed by standardization. Person, phone setup and movement execution were changed together, so the loss in accuracy cannot be explained by the participant difference alone.

Lab 4 results on another person



An accuracy of 99.19% was obtained on the controlled test, but approximately 25.78% was obtained on the other person. A decrease of 73.42 percentage points was measured. Jumping was predicted for 9,440 of the 9,656 external windows, or 97.76%. Almost all of the estimated jumping section was recognized. No walking windows and only 56 running windows were classified correctly.

Three-position stepping was performed as the other person's fourth activity, but forward/backward movement had been used for training. This mismatch is shown in the last row. The 25.78% score was calculated using the notebook's approximate four-section mapping; a matched fourth-class evaluation was not provided. A lower external score was obtained than in Lab 2, but a fair ranking of the two models cannot be made from these different tests. Reliable recognition across people was not demonstrated by either model.

Model choice limitations and future work

Final model choice

The compact GRU would be selected for the original controlled task. The highest controlled test accuracy and much better walking recognition were obtained with this model. No improvement was obtained by using a larger recurrent model, so the one-layer model with 32 units would be retained. KNN would also be kept as a useful baseline. No test advantage over KNN was obtained with the feedforward model.
For use across people or in natural continuous activity, further testing would be required before any of the models could be selected for reliable use. The GRU would be kept as the main candidate because of its controlled result, while its continuous and external failures would be investigated.

Main limitations

The development data were collected from one person, with only 12 recordings per class. Limited variation in movement style, speed, sessions and phone setup was included. Similar collection conditions were used for the controlled test. Recognition on new people and in new conditions could not be established from that test alone. Only one external person was included in each external evaluation.
Consistent phone orientation and movement definitions were not maintained across all evaluations. In Lab 4, the external fourth activity was different from the trained class. Approximate time boundaries were used for continuous activities. Highly overlapping windows were counted, although neighbouring predictions were strongly related. Class failures were also hidden when only overall accuracy was considered. High confidence was obtained for some incorrect predictions.

What would be improved

First, balanced data would be collected from several people across different sessions. The four movements would be defined in the same way for everyone. Phone placement and orientation would be recorded, and exact activity-change times would be marked. Whole participants and sessions would be kept separate for validation and final testing.
Second, gravity alignment, sensor magnitudes and rotation-based data augmentation would be tested to reduce the effect of phone orientation. The compact GRU would be compared with KNN using window features and with a feedforward model using the same windows. Time order would also be shuffled in a separate comparison so that its contribution could be checked.
Third, lower window overlap would be tested. Class recall, macro F1, recording-level results and transition results would be reported alongside accuracy. Training would be repeated with different random seeds. Confidence calibration and rejection of unfamiliar inputs would also be investigated.

Overall conclusion

A progression from single-row classification to sequence modelling was completed across the lab series. Better controlled recognition was achieved when temporal information was included. However, high controlled accuracy was not followed by reliable recognition on continuous or other-person data. The main next step should therefore be improved data collection and more realistic evaluation, rather than a more complex model.