Розпізнавання фізичної активності за допомогою моделі нейронної мережі

Постановка задачі:

1) Назбирати дата сет для декількох класів фізичної активності (збалансований), маркування даних і дата аналіз.
2) Використати декілька алгоритмів по feature selection і порівняти результати, які вони будуть повертати;
3) Поставити "правильні" активаційні функції для власних моделей;
4) Вивести метрики, порівняти результати метрик, зробити висновки;
5) Все на базі методологій;
6) Глянути до методів tensorflow чи pytorch.

Критерії валідації:

1) Точність моделі нейронної мережі 80% і більше.

Етап 1: Data Collection

Дані було зібрано за допомогою програми HyperIMU. Налаштування параметрів запису даних наведено на рис. 1 та рис. 2.

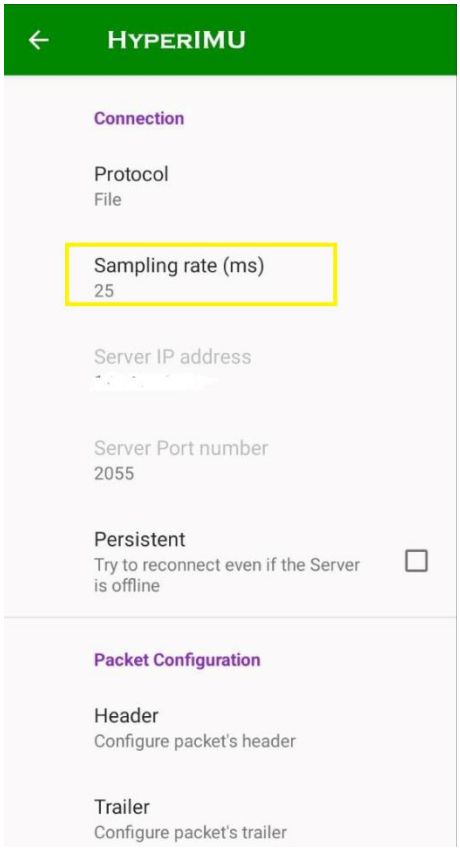

Рис. 1. Параметри запису даних акселерометра та гіроскопа

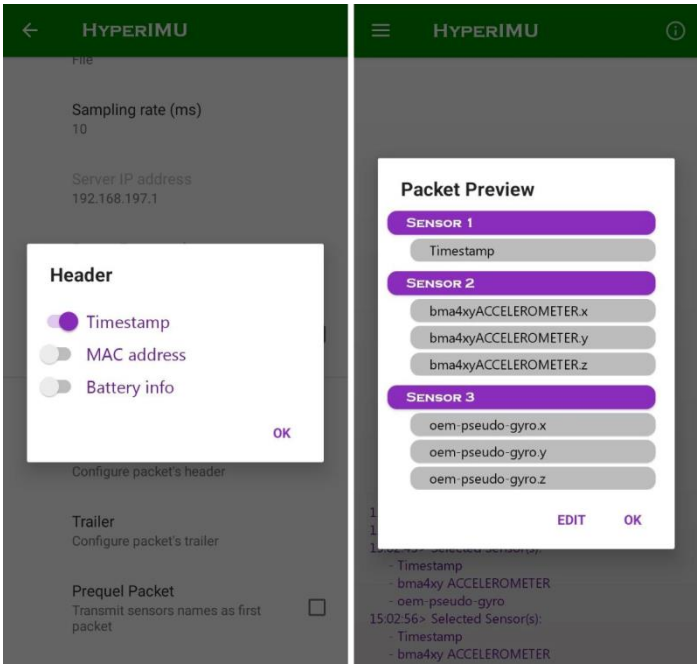

Рис. 2. Інформація про дані, які будуть записуватися

Із рис. 1 видно, що період запису даних становить 25 мс, тобто частота запису даних дорівнює $1/25 * 10^{-3} = 40$ Гц.

Окрім даних з осей акселерометра та гіроскопа було також додано стовпець 'Timestamp', який потрібен буде для дослідження стабільності частоти забору даних.

Дані записувалися для п’яти різних видів фізичної активності:

1) Присідання;
2) Ходьба;
3) Легкий біг (Jogging);
4) Випади на кожну ногу (рис. 3);
5) Lateral Squat Slide (Переміщення центру маси з однієї ноги на іншу) (рис. 4).

Рис. 3. Випади на кожну ногу

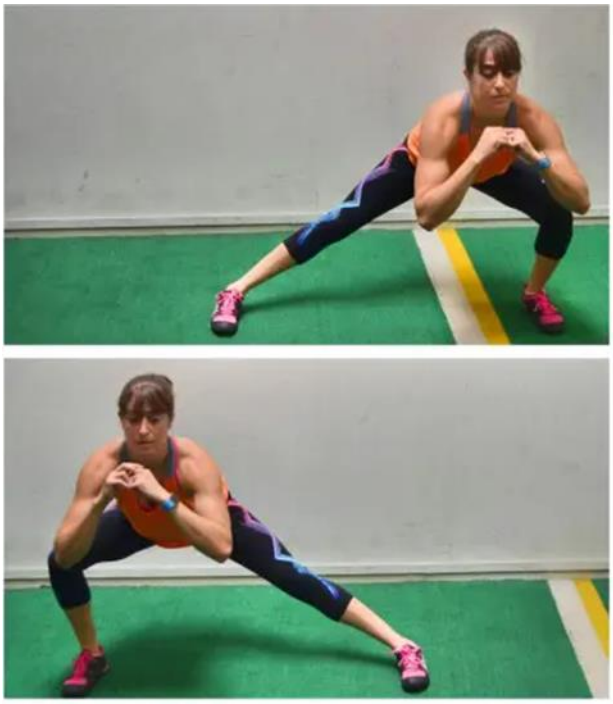

Рис. 4. Lateral squat slide

Під час запису даних смартфон знаходився у правій руці у горизонтальному положенні (вісь OZ акселерометра смартфона спрямована перпендикулярно до горизонталі).

Для кожного виду активності записувався окремий датасет/-и тривалістю більше 5 хв. Зокрема:

1) Присідання – 5 датасетів тривалістю понад 1 хв (приблизно 1 хв 15 с) кожен;
2) Ходьба – 1 датасет тривалістю 5.5 хв;
3) Легкий біг – 2 датасети тривалістю приблизно по 3 хв;
4) Випади на кожну ногу – 5 датасетів тривалістю приблизно по 1 хв 20 с кожен;
5) Lateral Squat Slide – 2 датасети тривалістю 1 хв 15 с + 2 датасети тривалістю приблизно 1 хв 45 с.

Усі вищезгадані датасети було об’єднано в один-єдиний тренувальний датасет.

Також для тестування було записано окремий датасет, який містить дані всіх видів активності (вправи виконувалися відразу одна після одної із невеликими перервами).

Отже, записані датасети містять сім стовпців: перший містить дані 'timestamp' у мс, наступні три відповідають даним трьох осей (OX, OY, OZ) акселерометра і виміряні у $м/с^2$, останні три стовпці – три осі гіроскопа (рад/с).

Етап 2: Data Cleaning and Preprocessing

Оскільки датасети замірялися у різні дні, що було викликано неможливістю виконання вправ через втому м’язів, для дослідження стабільності частоти забору даних було створено окремий стовпець - 'time', який містить час здійснення заміру відносно початку виконання вправ (позначка 0) так, ніби згадані раніше вправи виконувалися безперервно в часі (як у випадку із тестовим датасетом). Для того, щоб об’єднати два датасети, різниця в часі між замірами яких є значною (наприклад, заміри для першого датасету присідань робилися о 13:20, а для другого – о 20:18 того ж дня), проміжок часу між двома датасетами замінявся середнім значенням періоду одного заміру першого датасету.

Також з метою запобігання проблем із маркуванням даних у Label Studio та тренування моделей нейронної мережі було здійснено декілька перевірок вмісту отриманого датафрейму:

1) Наявність null-values;
2) Наявність рядків із значенням стовпця 'time' < 0;
3) Час йде у правильному напрямку, тобто збільшується (перевірка наявності викидів у стовпці 'time').

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 63529 entries, 0 to 63528
Data columns (total 8 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   timestamp  63529 non-null  int64  
 1   time       63529 non-null  float64
 2   accX       63529 non-null  float64
 3   accY       63529 non-null  float64
 4   accZ       63529 non-null  float64
 5   gyrX       63529 non-null  float64
 6   gyrY       63529 non-null  float64
 7   gyrZ       63529 non-null  float64
dtypes: float64(7), int64(1)
memory usage: 3.9 MB
```

Рис. 5. Інформація про тренувальний датасет

```
len(df[df['time']<0])
```

```
0
```

Рис. 6. Перевірка наявності від’ємного часу

```
# Check if the 'time' column of the dataframe is sorted in ascending order
df['time'].is_monotonic_increasing
```

```
True
```

Рис. 7. Перевірка наявності викидів у стовпці 'time'

Аналогічні перевірки було здійснено і для тестового датасету (рис. 8).

```
test_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12557 entries, 0 to 12556
Data columns (total 8 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   timestamp  12557 non-null  int64  
 1   time       12557 non-null  float64
 2   accX       12557 non-null  float64
 3   accY       12557 non-null  float64
 4   accZ       12557 non-null  float64
 5   gyrX       12557 non-null  float64
 6   gyrY       12557 non-null  float64
 7   gyrZ       12557 non-null  float64
dtypes: float64(7), int64(1)
memory usage: 784.9 KB
```

```
len(test_df[test_df['time'] < 0])
```

```
0
```

```
test_df['time'].is_monotonic_increasing
```

```
True
```

Рис. 8. Перевірка вмісту датафрейму test_df (тестового датасету)

Проведені тести не показали наявності у датасетах аномалій або викидів, тому можна переходити до маркування даних у Label Studio.

Етап 3: Data Labeling

Тренувальний і тестовий датасети було промарковано у Label Studio (рис. 9-10).

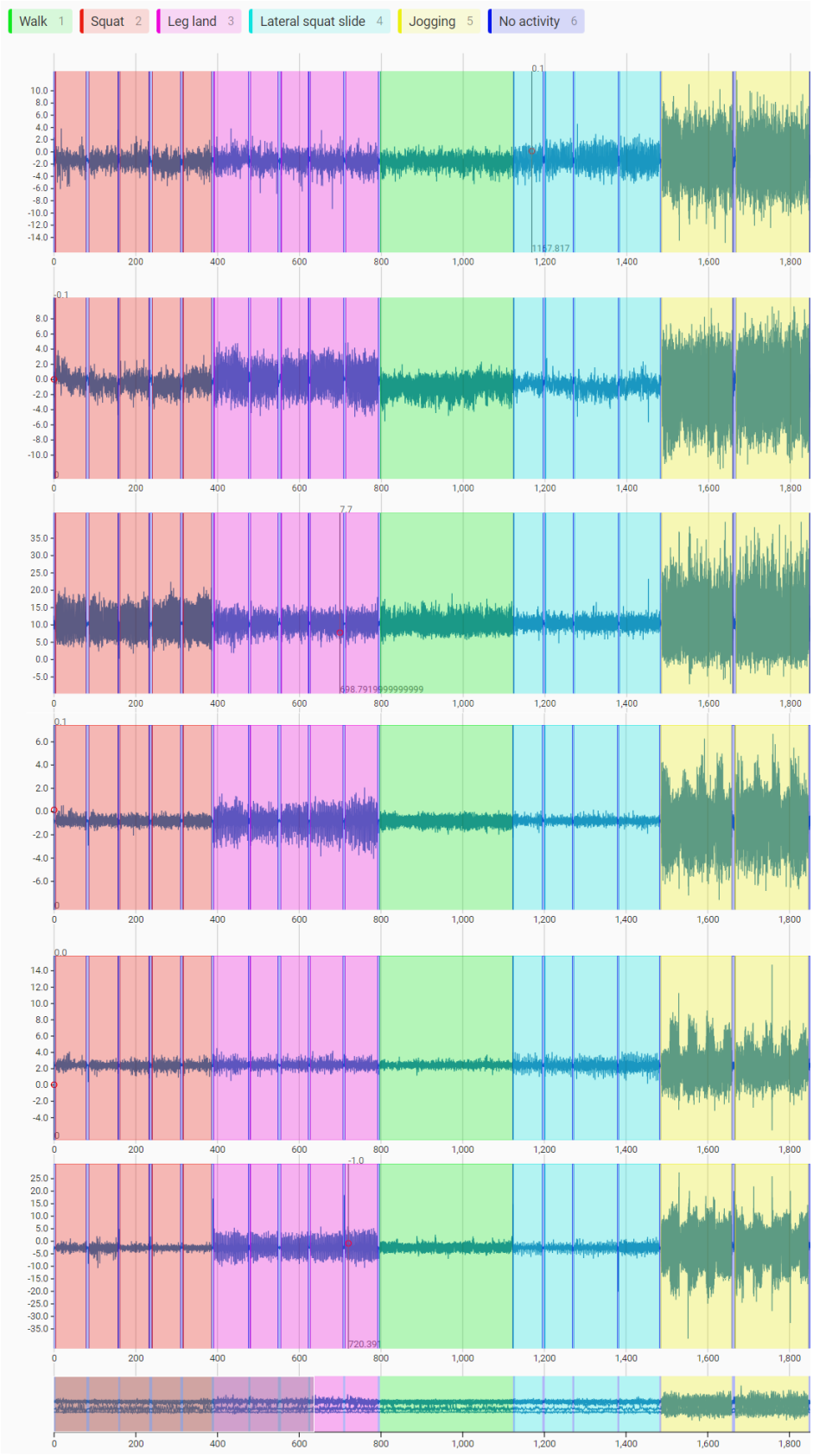

Рис. 9. Результати маркування тренувального датасету у Label Studio

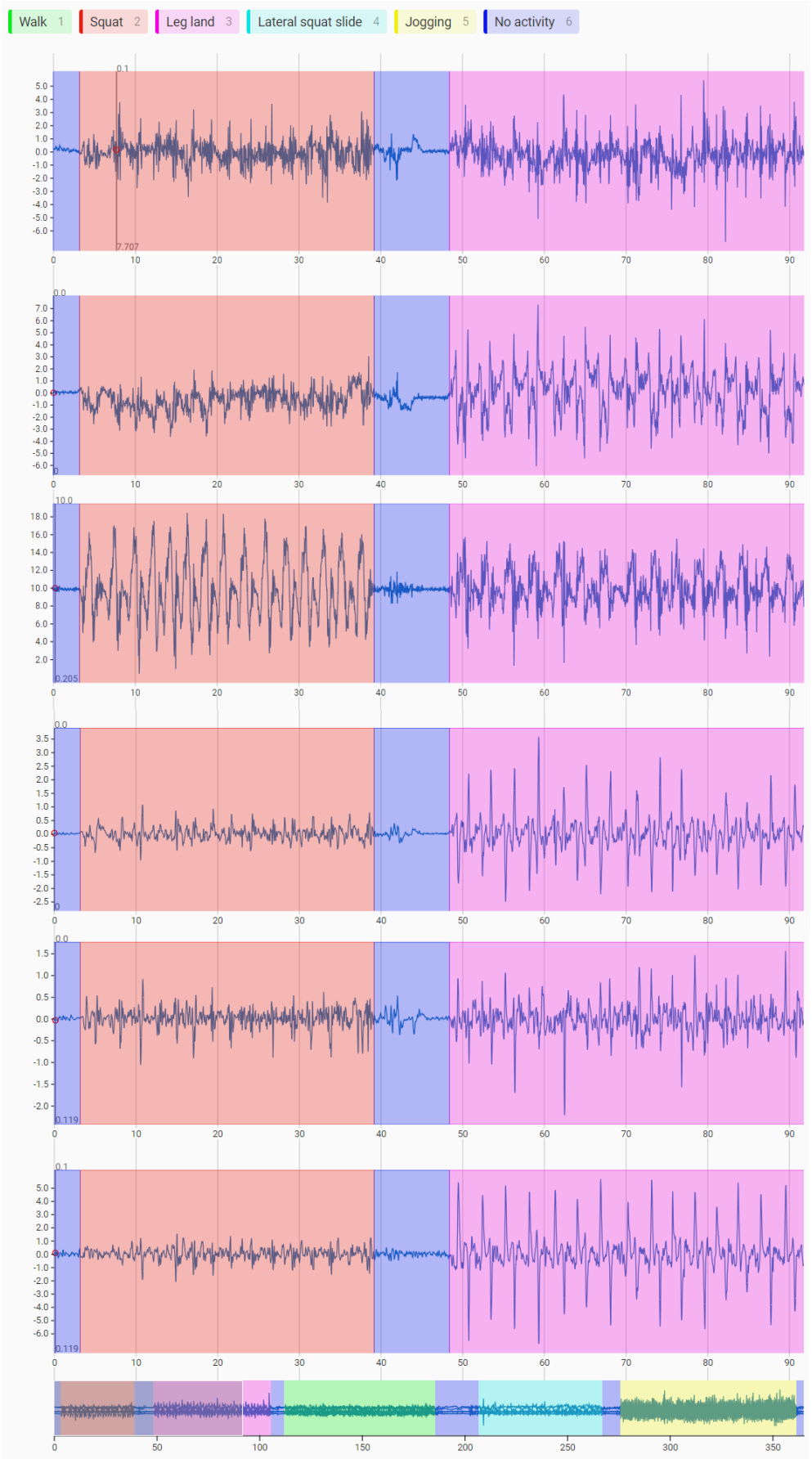

Рис. 10. Результати маркування тестового датасету у Label Studio

Варто відзначити наявність ще одного класу - 'No activity' (рис. 9-10), який використано для позначення калібрування датчиків на початку та в кінці запису даних (тренувальний датасет) та перерви між вправами (тестовий датасет), які були необхідні для відпочинку м’язів.

Після маркування даних до тренувального і тестового датасетів було додано ще один стовпець - 'activity', який містить 6 класів, наведених на рис. 9 і рис. 10.

Етап 4: Frequency stability analysis

Перед фільтруванням показів осей акселерометра та гіроскопа (Data Filtering), розглянемо спочатку стабільність частоти забору даних для тренувального і тестового датасетів.

```
time_diffs = df['time'].diff()
freq = 1.0 / time_diffs.mean()
print(f"Measurement time = {df.iloc[-1]['time'] - df.iloc[0]['time']} s")
print(f"Number of measurements (number of rows in the data set) = {len(df)}")
print(f"Average measurement period = {time_diffs.mean():.3f} s")
print(f"Average frequency of measurement = {freq:.3f} Hz")
```

```
Measurement time = 1852.281 s
Number of measurements (number of rows in the data set) = 63529
Average measurement period = 0.029 s
Average frequency of measurement = 34.297 Hz
```

Рис. 11. Інформація про час вимірювання, середній період та частоту забору даних тренувального датасету

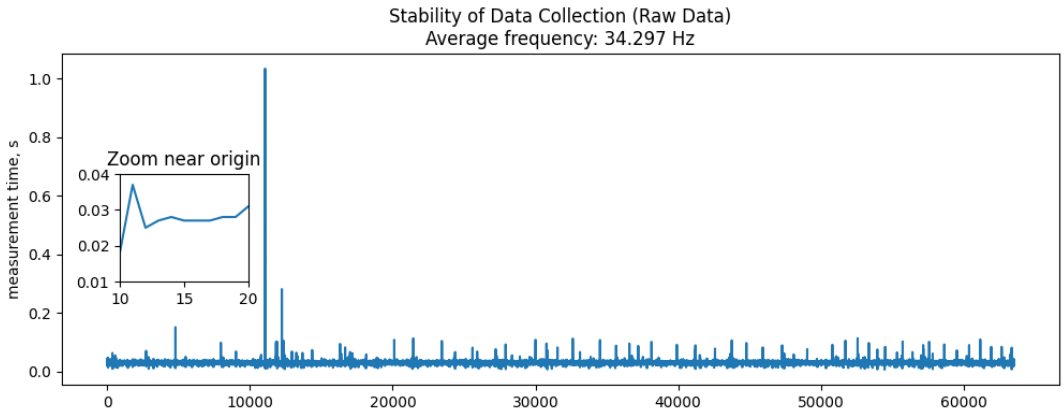

Рис. 12. Графік стабільності періоду забору даних тренувального датасету

Із рис. 12 видно, що більша частина даних замірялася із періодом приблизно 0.03 с. Однак спостерігаються і аномальні значення періоду вимірювання, серед яких яскраво вирізняється значення періоду понад 1 с.

Пояснити отримані коливання періоду забору даних можна так:

1) Незначні коливання відносно очікуваного значення періоду (25 мс) пояснюються наявністю обмеження апаратного забезпечення та точністю позначок часу, через що фактичні інтервали вибірки можуть дещо відрізнятися від запланованого значення. Невеликі варіації частоти дискретизації можуть призвести до запису точок даних з періодами трохи довшими за очікуване значення.
2) Наявність даних із періодом вимірювання понад 40 мс може бути пов'язана з перериванням процесу запису даних для виконання періодичних збережень або інших операцій. Зокрема, буферизація та очищення: для оптимізації збереження даних можна використовувати механізми буферизації для збору даних пакетами перед записом у сховище. Періодично ці буфери очищаються, що призводить до пауз у процесі запису даних.
3) Наявність аномалій із періодом запису понад 100 мс можна пояснити двома причинами:

   3.1) Фонові процеси: операційна система смартфона та інші фонові процеси можуть впливати на частоту дискретизації датчика. Якщо пристрій перебуває під великим навантаженням або працює з енергоємними програмами, це може вплинути на частоту дискретизації датчика.

   3.2) Помилки датчиків або збої: технічні проблеми, помилки датчиків або збої також можуть призвести до нерегулярних інтервалів вибірки в записаних даних.

   Обидва пояснення звучать реалістично, враховуючи, що деякі вимірювання проводилися із коротким інтервалом часу між вправами, що в свою чергу сприяло нагріванню процесора смартфона (особливо в жаркі літні дні) і тому могли відбуватися збої датчиків, а запис даних цілком міг перериватися системними процесами для охолодження пристрою.

Враховуючи майже однакову наявність аномалій та викидів (період вимірювання перевищує 40 мс) протягом усього часу вимірювання, було прийнято рішення видалити із оригінального датасету відповідні рядки, в результаті чого розмір датасету зменшився із 63529 (рис. 5) до 63049 (рис. 13).

```
df = df[df['time'].diff() <= 0.04]
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 63049 entries, 2 to 63528
Data columns (total 9 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   timestamp  63049 non-null  int64  
 1   time       63049 non-null  float64
 2   accX       63049 non-null  float64
 3   accY       63049 non-null  float64
 4   accZ       63049 non-null  float64
 5   gyrX       63049 non-null  float64
 6   gyrY       63049 non-null  float64
 7   gyrZ       63049 non-null  float64
 8   activity   63049 non-null  object 
dtypes: float64(7), int64(1), object(1)
memory usage: 4.8+ MB
```

Рис. 13. Вилучення аномалій, пов’язаних із періодом забору даних

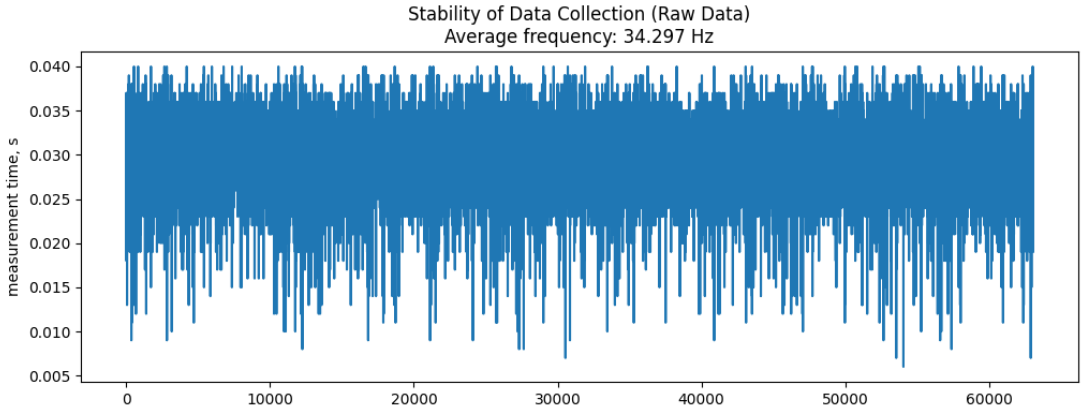

Рис. 14. Графік стабільності періоду забору даних після видалення аномалій та викидів

Розглянемо також стабільність частоти забору даних і для тестового датасету.

```
time_diffs = test_df['time'].diff()
freq = 1.0 / time_diffs.mean()
print(f"Measurement time = {test_df.iloc[-1]['time'] - test_df.iloc[0]['time']} s")
print(f"Number of measurements (number of rows in the data set) = {len(test_df)}")
print(f"Average measurement period = {time_diffs.mean():.3f} s")
print(f"Average frequency of measurement = {freq:.3f} Hz")
```

```
Measurement time = 365.243 s
Number of measurements (number of rows in the data set) = 12557
Average measurement period = 0.029 s
Average frequency of measurement = 34.377 Hz
```

Рис. 15. Інформація про час вимірювання, середній період та частоту забору даних тестового датасету

Порівнявши результати, наведені на рис. 11 і рис. 15, можна зробити висновок, що середній період і середня частота забору даних для тренувального і тестового датасету майже однакові.

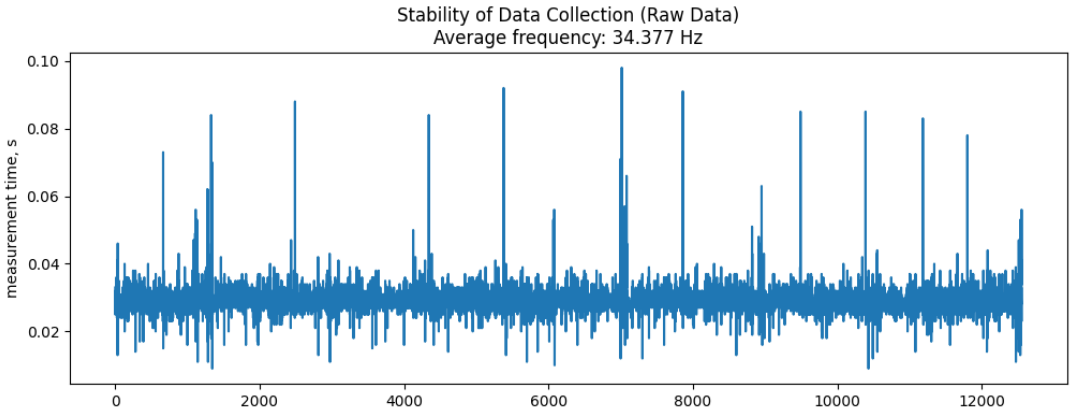

Рис. 16. Графік стабільності періоду забору даних тестового датасету

На графіку стабільності періоду забору даних (рис. 16) тестових даних все ще спостерігається наявність викидів (T>0.04 с), однак немає різких аномалій (понад 0.2 с, 1 с), як в тренувальному датасеті.

```
len(test_df[test_df['time'].diff() > 0.040])
```

```
84
```

Рис. 17. Кількість викидів у тестовому датасеті

Хоча кількість викидів, пов’язаних із стабільністю періоду забору даних, у тестовому датасеті є незначною (рис. 17) і їх можна було б видалити, для чистоти експерименту (щоб перевірити модель нейронної мережі на реальних, неопрацьованих даних), було прийнято рішення не вносити змін у тестовий датасет.

Етап 5: Data Filtering

Перед фільтруванням даних візуалізовано покази всіх осей акселерометра та гіроскопа оригінального тренувального датасету (рис. 18).

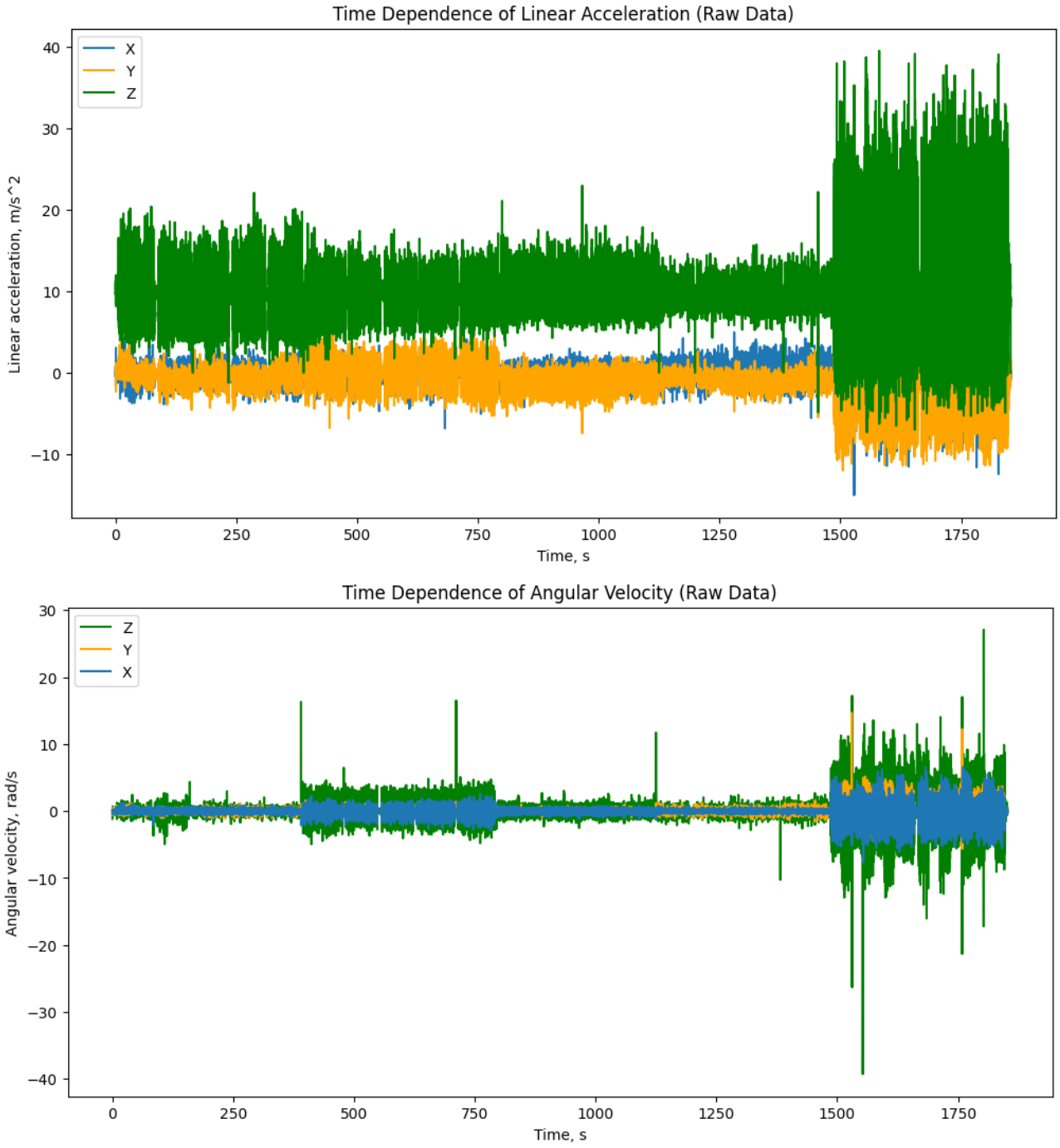

Рис. 18. Покази акселерометра та гіроскопа у тренувальному датасеті (невідфільтровані дані)

Для фільтрування даних використано медіанний фільтр із розміром вікна 10. Результати фільтрування результатів вимірювань акселерометра та гіроскопа тренувального датасету наведено на рис. 19.

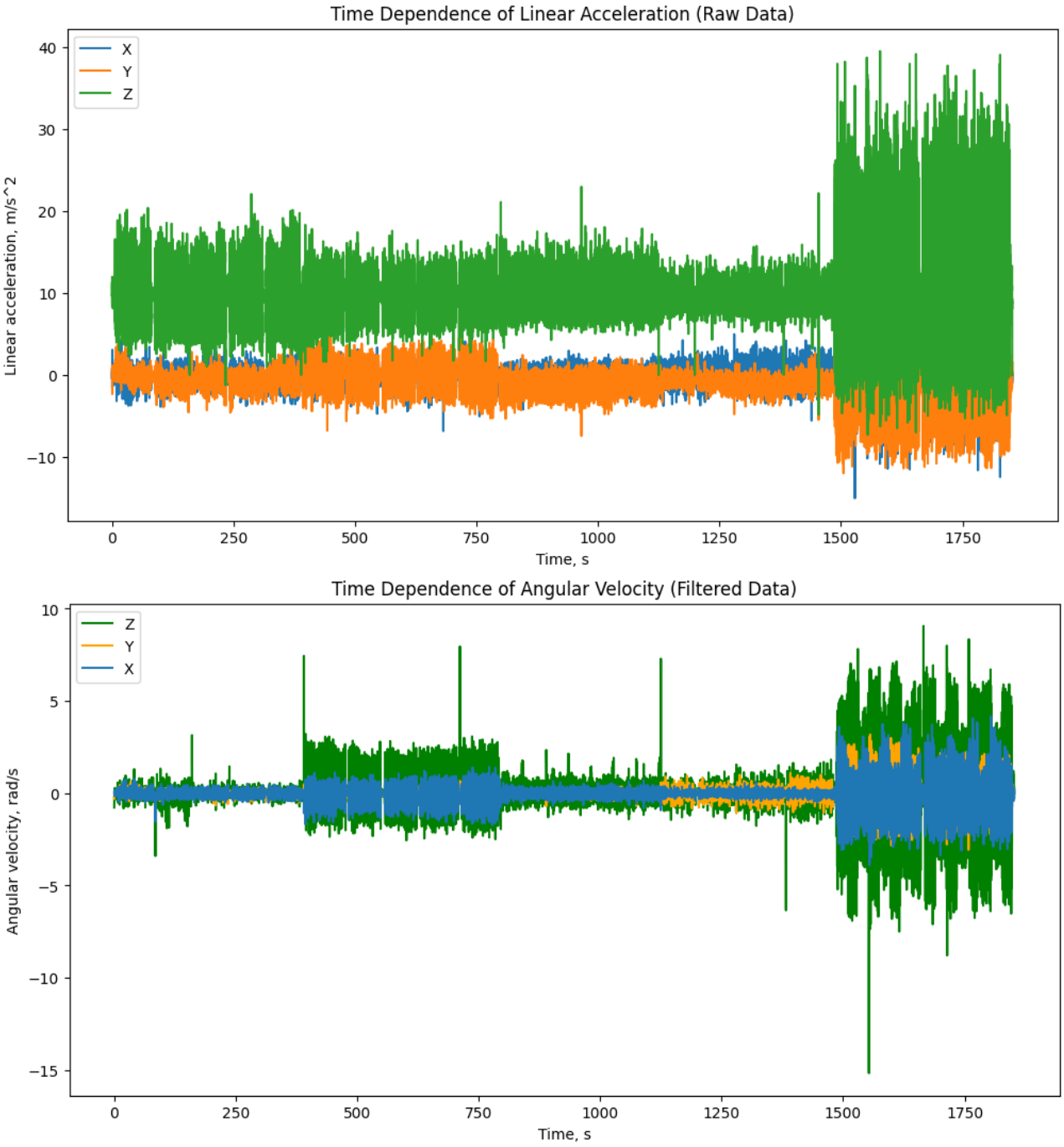

Рис. 19. Покази акселерометра та гіроскопа у тренувальному датасеті (відфільтровані дані)

Щоб нагляд ніше продемонструвати вплив медіанного фільтра, побудовано графіки для осі OX акселерометра (рис. 20) та осі OZ гіроскопа (рис. 21), на яких відображено невідфільтровані та відфільтровані дані.

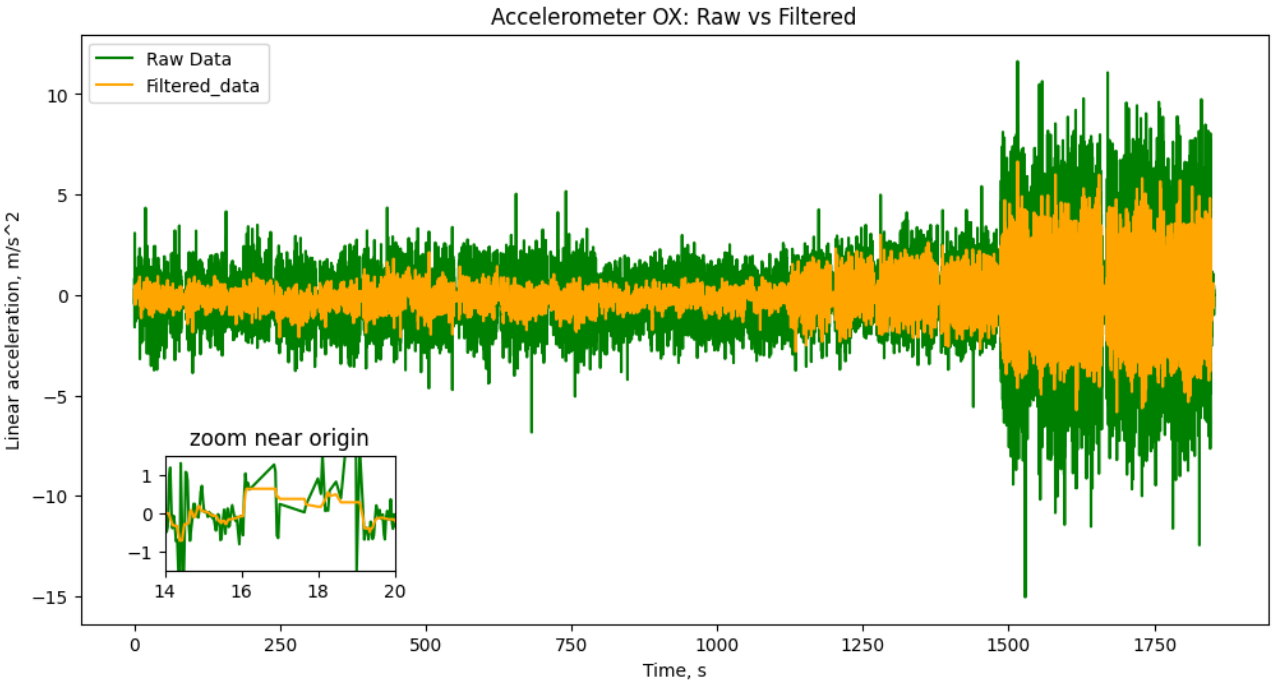

Рис. 20. Результат фільтрування осі OX акселерометра

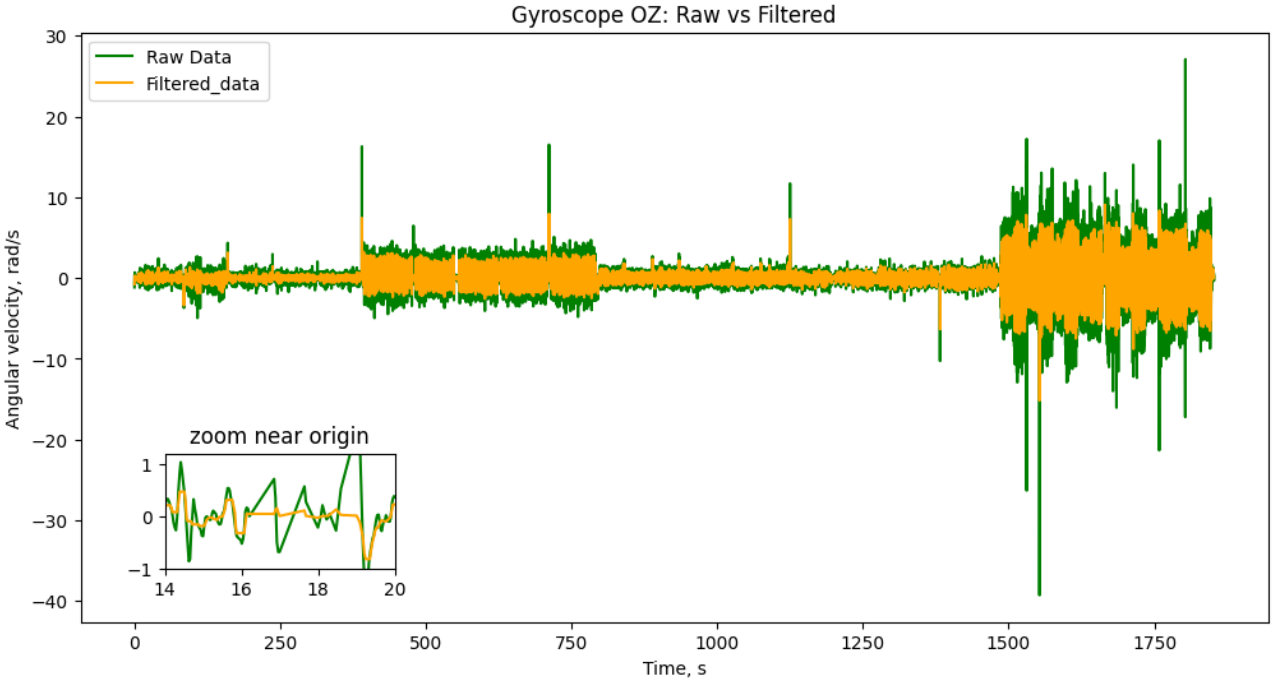

Рис. 21. Результат фільтрування осі OZ гіроскопа

Як видно із рис. 20 і рис. 21, внаслідок фільтрування відбулося згладження деяких викидів (особливо помітно на осі OZ гіроскопа), а результати загалом були усереднені. При цьому, загальні особливості для кожного виду фізичної активності збереглися.

Етап 6: Exploratory Data Analysis

Для початку проаналізуємо розподіл міток видів активностей, тобто розподіл цільових класів.

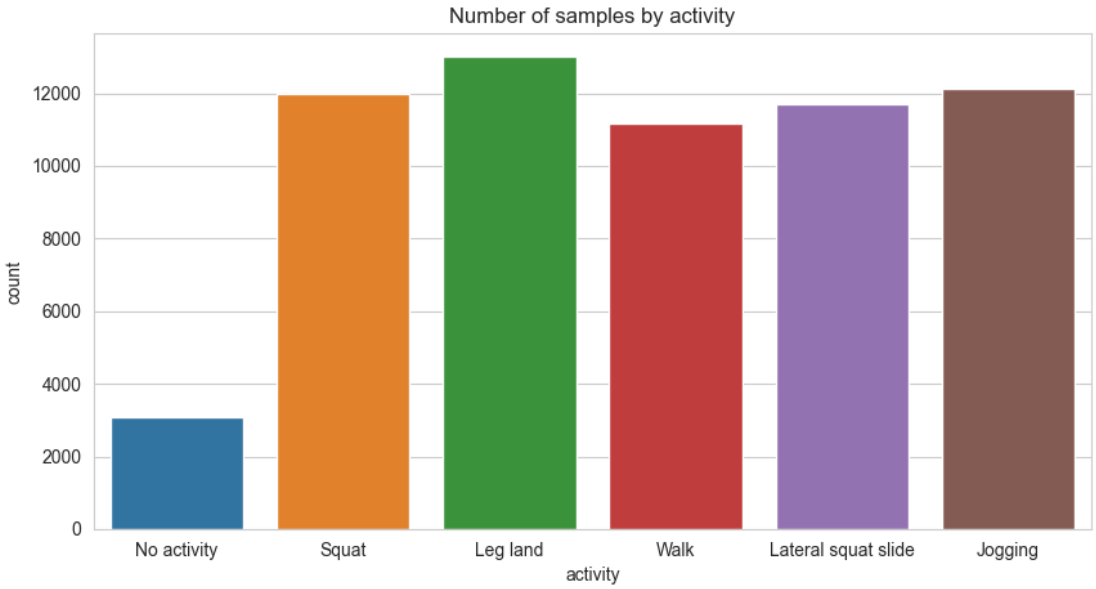

Рис. 22. Аналіз розподілу класів тренувального датасету

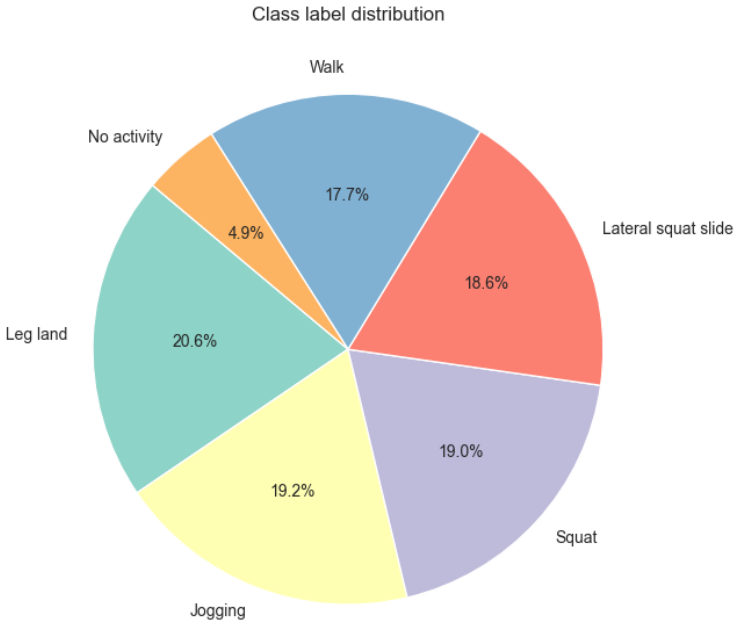

Рис. 23. Аналіз розподілу класів тренувального дата сету у процентному співвідношенні

Із рис. 22 і рис. 23 видно, що 5 досліджуваних видів фізичної активності (присідання, ходьба, легкий біг, випади на ноги та lateral squat slide) представлені майже однаковою кількістю елементів. Клас 'No activity', який

становить менше 5% обсягу датасету, не становить цінності для тренування моделі нейронної мережі, тому його можна вилучити.

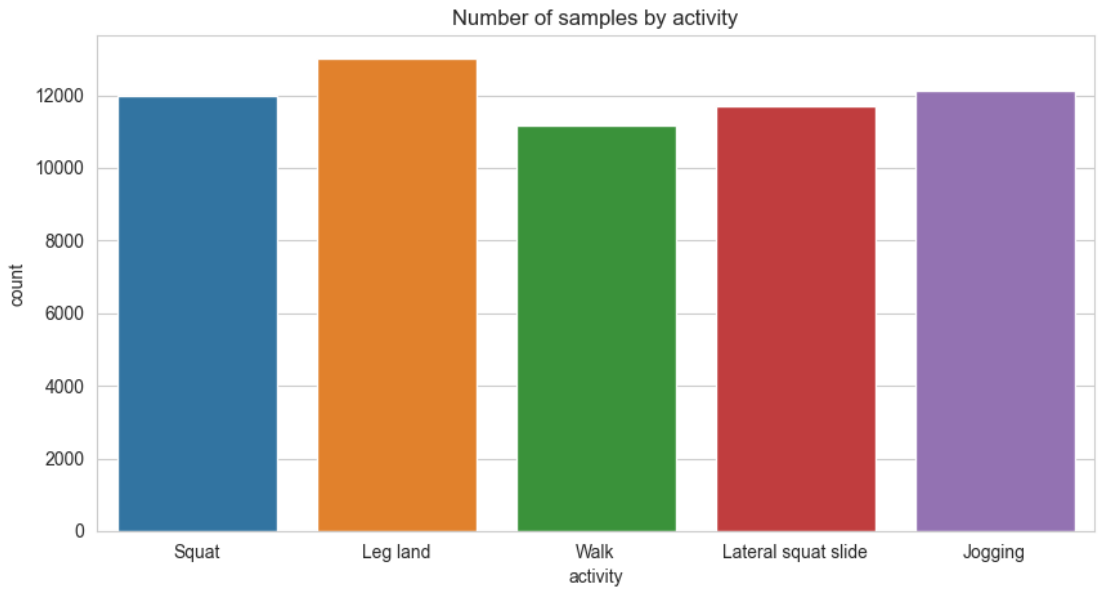

Рис. 24. Аналіз розподілу класів тренувального датасету після вилучення класу 'No activity'

Із рис. 24 видно, що кількість записів для кожного виду фізичної активності є майже однаковою. Однак, між випадами на ноги ('Leg land') та ходьбою ('Walk') спостерігається помітна різниця в кількості записів – майже 2000 (рис. 24, рис. 25).

```
activity_counts = df['activity'].value_counts()
print(activity_counts)
```

```
Leg land              13000
Jogging               12121
Squat                 11985
Lateral squat slide   11710
Walk                  11153
Name: activity, dtype: int64
```

Рис. 25. Кількість записів для кожного виду фізичної активності тренувального датасету

Зрівняємо кількість записів для всіх видів фізичної активності, використавши undersampling (рис. 26), відкинувши останні n записів для відповідного класу активності. Внаслідок виконання процесу undersampling, розмір тренувального датасету зменшився від 63049 (рис. 13) до 55765 записів (рис. 26). Натомість було отримано збалансований датасет, який містить 11153 записи для кожного класу фізичної активності (рис. 26-27).

```
undersampled_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55765 entries, 0 to 55764
```

```
activity_counts = undersampled_df['activity'].value_counts()
print(activity_counts)
```

```
Squat                  11153
Leg land               11153
Walk                   11153
Lateral squat slide    11153
Jogging                11153
Name: activity, dtype: int64
```

Рис. 26. Результат виконання undersampling для тренувального датасету

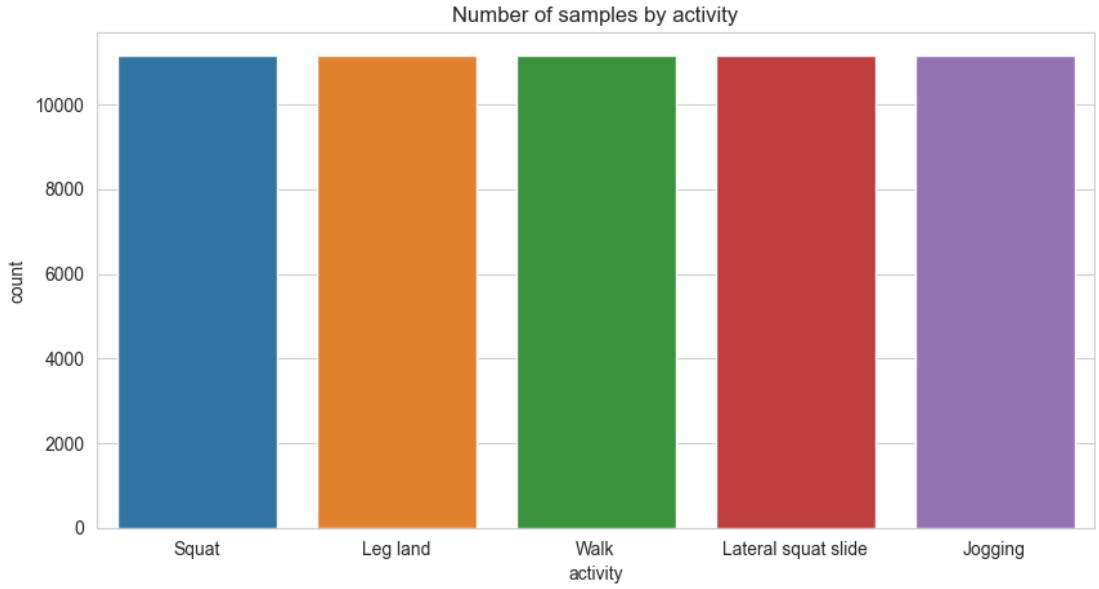

Рис. 27. Розподіл класів фізичної активності тренувального датасету після виконання undersampling