COMP90051 Statistical Machine Learning – 2021 Sem2
Project 1 Specification

Competition link: https://www.kaggle.com/t/627ca058065f4c62a708e6a15c05c758
Competion closes: 4pm Thu, 9th Sep; **Reports due:** 3pm Fri, 10th Sep. **Total Weight:** 25%

1 Overview

Human action recognition, i.e., recognizing the category of a human behaviour from a sequence, is very important in our daily life. Human action recognition techniques can be used in many applications such as detecting dangerous behaviours in visual surveillance and analysing pedestrian behaviours for safe operation in autonomous driving systems.

In real-world scenarios, some human behaviours (e.g., 'walking') are quite common. It is easy to collect many samples for such actions. However, there are also some other behaviours (e.g., dangerous actions) that are less common. In such cases, it is hard to collect many samples for such actions. In other words, collected action recognition datasets may suffer from the problem of class imbalance.

In this case, it is very important to handle the class imbalance problem when training an algorithm for action recognition.

Your task:

In this project, you will be learning from a training dataset (9388 sequences) with class imbalance and trying to recognise the labels of a testing dataset (2959 testing sequences). In each sequence, there are 16 frames. Each frame contains the features of human skeleton joints that can be used to model the action. More details of the dataset are provided in Section 2.

To make the project fun, we will run it as a Kaggle in-class competition. Your assessment will be partially based on your final ranking in the privately-held competition, partially based on your absolute performance and partially based on your short report.

2 Data Format

All data will be provided in CSV files. Each row corresponds to one sequence sample, with sample ID, features of all frames and label ID in the training file, and sample ID and features of all frames in the testing file. Each sequence contains 16 frames. Each frame contains the motion features of 20 skeleton joints of a human body as shown in Figure 1. More specifically, each joint is represented with features in x, y, z axes in 3D space. Thus each frame contains $20 \times 3 = 60$ features. A sequence with 16 frames contains $20 \times 3 \times 16 = 960$ features, so each row in the training CSV file contains 962 columns (1 sample ID + 960 features + 1 label ID) and each row in the testing CSV file contains 961 columns (1 sample ID + 960 features). The features in each row in the CSV files are organised as:

```
Frame1.Joint1.x Frame1.Joint1.y Frame1.Joint1.z ...  Frame1.Joint20.x Frame1.Joint20.y Frame1.Joint 20.z
Frame2.Joint1.x Frame2.Joint1.y Frame2.Joint1.z ...  Frame2.Joint20.x Frame2.Joint20.y Frame2.Joint 20.z ...
Frame16.Joint1.x Frame16.Joint1.y Frame16.Joint1.z ...  Frame16.Joint20.x, Frame16.Joint20.y, Frame16.Joint 20.z
```

There are 9388 training sequences in total, that belong to 49 classes (1 to 49). The number of samples in each class ranges from 590 to 24. There are 2959 testing sequences. Your implemented algorithm should take the test sequences in and return a 2-column 2960-row CSV file that has (a) in the first (header) row, the string "`ID,Category`"; (b) in all subsequent rows, a consecutive integer and a comma, then an integer in the range [1,49], which is your predicted class ID of each testing sequence.

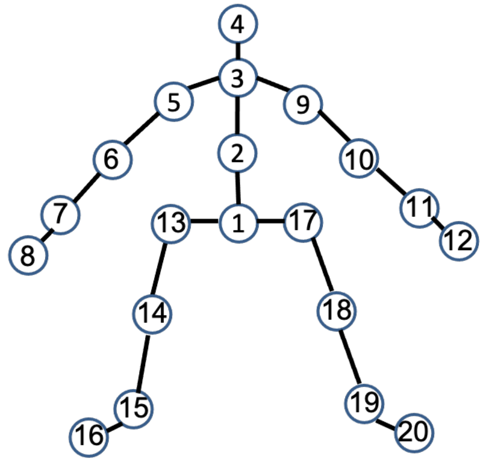

Figure 1: 20 human skeleton joints, including (1) base of the spine, (2) middle of the spine, (3) neck, (4) head, (5) left shoulder, (6) left elbow, (7) left wrist, (8) left hand, (9) right shoulder, (10) right elbow, (11) right wrist, (12) right hand, (13) left hip, (14) left knee, (15) left ankle, (16) left foot, (17) right hip, (18) right knee, (19) right ankle, (20) right foot.

For example, given two testing sequences with sample ID 1 and 2, your predicted classes for the two testing sequences are 43 and 22, respectively, then your output file should be:

```
Id,Category
1,43
2,22
```

The test set will be used to generate an accuracy (**ACC**) for your performance. You may submit test predictions up to 10 times per day (if you wish). During the competition ACC on a 30% subset of the test set will be used to rank you in the **public leaderboard**. We will use the **private leaderboard** (70% test set) to determine your **final ACC and ranking**. The split of test set during/after the competition is used to **discourage overfitting the test set**. The training CSV file "train.csv", the testing CSV file "test.csv" and a sample submission file "sample.csv" will be available within the Kaggle competition website. In addition to using the competition testing and to prevent overfitting, we encourage you to generate your own test sets from the training set, i.e., **a validation set**, to develop, model select and test your algorithms. Doing so is good practice, and the setup of this project encourages this.

Students are not permitted to augment the project data with external data. Doing so could result in a zero on the assessment.

3 Student Groups

You are expected to work in **groups of 3 students**. We will mark all teams based on our expectations of what a typical team could achieve: you might consider roles such as researcher, feature engineering, learning, workflows/scripting, experimentation, ensembling of team models, generating validation data, etc. and divide your identified roles among your team. As indicated on Canvas ahead of project release, you can either find team mates yourself, we can randomly assign team mates, or both (e.g. 2 with 1).

We encourage active discussion among teams of **conceptual ideas**, but sharing of code, data, or specific solutions are not allowed. Given your marks are partially dependent on your final ranking in the competition, it is in your interest not to collude.

Expected Group Behaviour. We expect all students to contribute equally to their groups, to be communicative, and at all times respectful of fellow students. Your peers will be your future professional networks and possibly co-workers. Group work is an important feature of all professional machine learning workplaces, and a key generic skill developed by this project. As described below, we require submission of a 'Group Agreement' and at least 3 group meetings with recorded 'Meeting Minutes' during the course of the project. These measures promote positive group dynamics with transparency around group expectations. From past experience of classes of this size, we expect at most a tiny handful of groups to have issues working together.

4 Submissions & Important Deadlines

Submissions in part are due on Kaggle via your Kaggle Team, and on Canvas via your Project Group. You should not submit to Kaggle via individual accounts to circumvent the daily restrictions.

Tue Aug 24, 10AM (optional). If you intend to form your own team in full (3 students), or in part (2 students), you must together join one "Project 1 group" on Canvas. Within a day of this point, we will randomly assign remaining students to groups including completing partial groups of 2.

Fri Aug 27, 6PM. Submit on Canvas your "Group Agreement" through your Canvas group. You are expected to have met with your team members by now; you should together discuss expectations of the group, so that you embark on the project with a shared set of goals and supporting roles. While this agreement is not marked, we require students to complete this.

Thu Sep 9, 4pm. Kaggle competition closes. You must ensure you have made at least one valid submission to the Kaggle in-class competition linked above. Kaggle will allow you to select up to 2 submissions to be used for your scoring.

Fri Sep 10, 3pm. The remainder of your project submission is due:

- PDF final report (see Section 5 for requirements) submitted through your Canvas group.
- Text of your Kaggle team name only. E.g. if your team name on Kaggle is *TeamStatML* then this submission should only have the text *TeamStatML*. If you add additional text, or submit something else, we may not be able to match your Kaggle submission to your team and you could receive zero for your competition performance.
- Zip file of supporting evidence: 3 (or more) meeting minutes following the template on Canvas; code that covers your class-imbalance action recognition algorithm in any language including scripts and automation, and a README. **DO NOT INCLUDE DATA** (to not overload Canvas); any further evidence of team work such as chat/git logs if desired. While we will not mark your code, we may use this information to verify your solution is your own work.

The submission link will be visible in Canvas prior to deadline.

Plagiarism policy: You are reminded that all submitted project work in this subject is to be your own individual team work. Automated similarity checking software will be used to compare submissions. It is University policy that academic integrity be enforced. For more details, please see the policy at `http://academichonesty.unimelb.edu.au/policy.html`. You may use software libraries and code snippets found online with attribution, but not code of other University of Melbourne students outside your group. As a reminder, you are not permitted to use any data beyond the data we provide you.

5 Report Requirements

Your final report should provide the following sections:

1. A brief description of the problem and introduction of any notation that you adopt in the report.
2. Description of your final approach(s) to address class-imbalance action recognition from sequential data, the motivation and reasoning behind it, and why you think it performed well/not well in the competition.
3. Any other alternatives you considered and why you chose your final approach over these (this may be in the form of empirical evaluation, but it must be to support your reasoning - examples like "method A, got ACC 0.2 and method B, got ACC 0.25, hence we use method B", with no further explanation, will be marked down).

Your description of the algorithm should be clear and concise. You should write it at a level that a postgraduate student can read and understand without difficulty. If you use any existing algorithms, *please do not rewrite the complete description, but provide a summary* that shows your understanding and references to the relevant literature. In the report, we will be interested in seeing evidence of your thought processes and reasoning for choosing one algorithm over another.

Dedicate space to any interesting details about software setup or your experimental pipeline, and any problems you encountered and what you learned. In many cases these issues are at least as important as the learning algorithm, if not more important.

Report format rules. The report should be submitted as a PDF, and be **no more than three single-column A4 pages**. The font size should be **11pt or above**. If a report is longer than three pages in length, we will only read and assess the report up to page three and ignore further pages. (Don't waste space on cover pages.)

6 Assessment

The project will be marked out of 25 and contribute 25 percent towards your subject total mark. **No late submissions accepted. You must inform your lecturers about sickness well before the deadline. Submit early and often to guard against unexpected last minute issues.**

The assessment in this project will be broken down into two components. The following criteria will be considered when allocating marks.

Kaggle Competition (13/25):

Your final mark for the Kaggle competition is based on your rank in that competition. Assuming N teams of enrolled students compete, there are no ties and you come in at R place (e.g. first place is 1, last is N) with an ACC of $A \in [0, 1]$ then your mark is calculated as

$$10 \times \frac{\max\{\min\{A, 0.5\} - 0.02, 0\}}{0.48} + 3 \times \frac{N - R}{N - 1} \ .$$

Ties are handled so that you are not penalised: tied teams receive the rank of the highest team (as if no team were tied). For example, if teams A, B, C, D, E came 1st, 4th, 2nd, 2nd, 5th, then the rank-based mark terms (out of 3) for the five teams would be 3, 0.75, 2.25, 2.25, 0.

This complicated-looking expression can result in marks from 0 all the way to 13. We are weighing more towards your absolute ACC than your ranking. Considering that this task is very challenging, we are also truncating your ACC at 0.5 — this score gives you the max possible contribution from the ACC (but ranking higher can still help you).

The component out of 10 for ACC gives a score of 0/10 for ACC of 0.02 or lower; 10/10 for ACC of 0.5 or higher; and linearly scales over the interval of ACCs [0.02, 0.5]. We believe that much higher than 0.02 ACC (random classifier) is achievable with minimal work, while 0.5 ACC is an excellent result deserving of full marks. *For example, a ACC of 0.3 for a team coming last would yield 5.8/13; or 7.3/13 if coming mid-way in the class.*

The rank-based term encourages healthy competition and discourages collusion. The other ACC-based term - rewards teams who don't place in the top but none-the-less achieve good absolute results.

Note that invalid submissions will come last *and* will attract a mark of 0 for this part, so please ensure your output conforms to the specified requirements.

Report (**12/25**):

The below marking rubric outlines the criteria that will be used to mark your report.

Critical Analysis (Maximum = 8 marks)	**Report Clarity and Structure** (Maximum = 4 marks)
8 marks Final approach is well motivated and its advantages/disadvantages clearly discussed; thorough and insightful analysis of why the final approach works/not work for provided training data; insightful discussion and analysis of other approaches and why they were not used	4 marks Very clear and accessible description of all that has been done, a postgraduate student can pick up the report and read with no difficulty.
6.4 marks Final approach is reasonably motivated and its advantages/disadvantages somewhat discussed; good analysis of why the final approach works/not work for provided training data; some discussion and analysis of other approaches and why they were not used	3.2 marks Clear description for the most part, with some minor deficiencies/loose ends.
4.8 marks Final approach is somewhat motivated and its advantages/disadvantages are discussed; limited analysis of why the final approach works/not work for provided training data; limited discussion and analysis of other approaches and why they were not used	2.4 marks Generally clear description, but there are notable gaps and/or unclear sections.
3.2 marks Final approach is marginally motivated and its advantages/disadvantages are discussed; little analysis of why the final approach works/not work for provided training data; little or no discussion and analysis of other approaches and why they were not used	1.6 marks The report is unclear on the whole and the reader has to work hard to discern what has been done.
1.6 marks Final approach is barely or not motivated and its advantages/disadvantages are not discussed; no analysis of why the final approach works/not work for provided training data; little or no discussion and analysis of other approaches and why they were not used	0.8 marks The report completely lacks structure, omits all key references and is barely understandable.