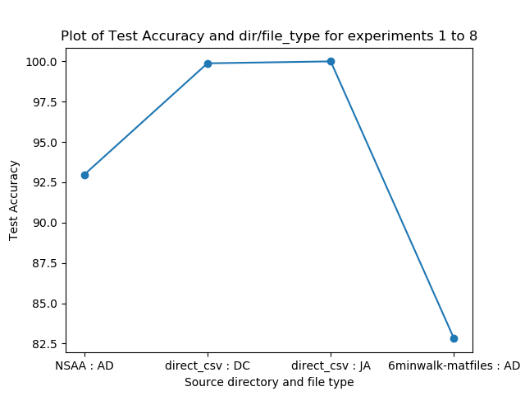
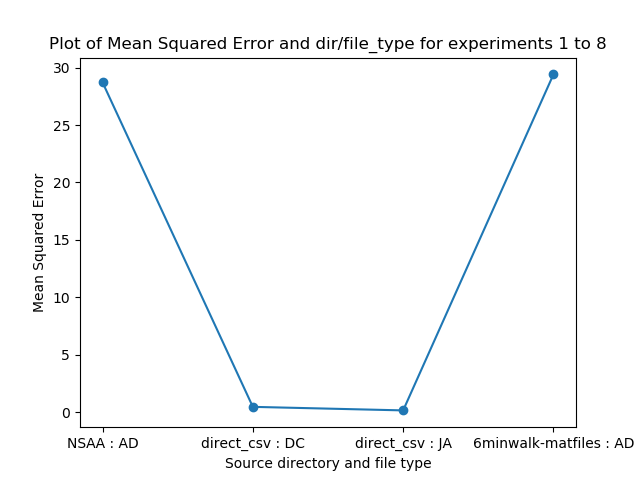
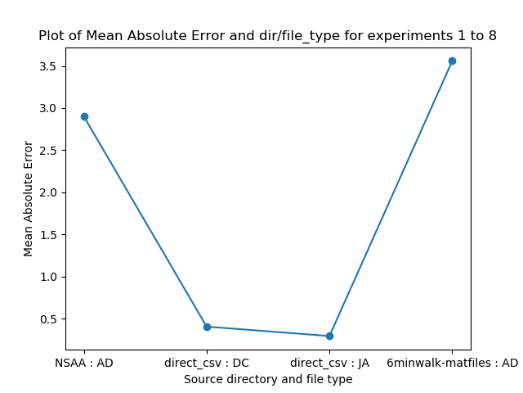
**Experiments and Results Discussion**

**Experiment Set 1: Performance of RNNs on Different Source Data**

The purpose of this experiment set is to determine whether and how well RNN models regresses on different types of source data. By this, we mean data that is in the same format (i.e. ‘.mat’ files in the same organizational structure) but representing different measurements from different source directories. These are thus:

* **NSAA : AD** – statistical values extracted from the ‘all-data’ (‘AD’) files by the ‘comp\_stat\_vals.py’ script, whose files are sourced from the ‘NSAA’ directory; hence these are stat values of the subjects performing the NSAA activities, which are then used to train model(s).
* **direct\_csv : DC** – these are the joint angle values (i.e. raw measurements, not computed stat values) that are sourced from the data cube; this data cube contains the 6-minute walk data from various subjects, several of which aren’t included in the standard joint angle files of subjects doing the 6-minute walk.
* **direct\_csv : JA** – similar to ‘direct\_csv : DC’ as described above, using the same source directory type and raw measurement, but containing joint angle files that aren’t necessarily included within the data cube.
* **6minwalk-matfiles : AD** – again, uses the files corresponding to subjects’ 6-minute walk assessments, with the difference this time being that we aren’t using raw joint angle files either in the data cube format or as ‘loose’ files, but rather computing statistical values via ‘comp\_stat\_vals.py’ script; in this sense, it’s the same as ‘NSAA : AD’ models but using different assessment data (6-minute walk rather than NSAA).

For the second and third diagrams above, with regards to the output type, all these file types and directory sources outlined above are used to train models trained to regress on the overall NSAA score for a given sequence from a file. This value is able to range between 0 and 34 (though based on how the assessment is done, it generally ranges between 15 and 34). Hence, for a given type of source data, if it has a MAE = 0.5, that means that, on test data of sequences from files of the given type of source data, the model predicts for each of them a score of between 0 and 34 on average ‘0.5’ away from the true value for that sequence (the true value for a sequence being the overall NSAA score of the file that the sequence comes from). However, for the first diagram above, it tests the different files types on its ability to classify whether a sequence that come from a file are from a ‘D’ or ‘HC’ subject.

**Results Discussion and Conclusions**

In using just the raw joint angle values from ‘DC’ or ‘JA’ files, we achieve an approximate **99%** accuracy (i.e. for each sequence of 60 rows of 66 joint angle values, the model can predict with 99% accuracy whether it came from a ‘D’ file or an ‘HC’ file); however, looking at more measurements (e.g. position, accelerometer values, etc.), performing manual feature extraction via computing of the statistical values and then reducing the dimensionality, and then training the model provides a much worse classification accuracy of **82.81%** for data that comes from the same assessment (6-minute walk). This can also be seen when the same data sources are then used to train the RNN to perform regression for the overall NSAA score: the raw joint angle data gives a much smaller **MAE = 0.4037** (meaning that it predicts a score of between 0 and 34 which on average is 0.4037 away from the true value in either direction), compared with a much worse **MAE = 3.56** from 6-minute walk ‘AD’ statistical value files. A further observation can be made about the experiments concerning the raw joint angle files in that they were performing much better than we were expecting them to be: by simply considering only the joint angle measurement of a subjects suit data, given 1 second’s worth of an input sequence to the RNN, it can correctly classify whether the frame comes from a healthy control subject or one with DMD to a very high accuracy of 99% and predict the overall NSAA score to within 0.4037 of the true value of between 0 and 34. This is extraordinarily high, much better than the ability of medical professionals and, most notably, this is only the first iteration of the experiments with raw measurement files.

This large difference between raw measurements and computed statistical features may seem counterintuitive at first glance, as the former is just using data direct from the provided ‘.mat’ files, while with the latter we actually process it further to ideally extract more important features from the data. One possible reason for raw joint angle models performing so well could be that there aren’t that many subjects with these files that are provided to us in comparison with ‘AD’ files, so not as diverse a training set is used; this means that there would be a narrower range of overall NSAA scores to regress towards, making it easier for the model to approximate the true score with a smaller margin of error. Another possible reason could be that the features that the RNN models extracts from its inputted data are more useful for it to train on and approximate an overall score: a prominent benefit of using neural networks is that they are traditionally noted to perform better with raw data rather than manually extracted features. Furthermore, when it trains on the stat value data from ‘comp\_stat\_values.py’ for the ‘AD’ file types, it still computes its own features within the RNN, and so this ‘features from features’ behaviour might have proved to be problematic for the network.

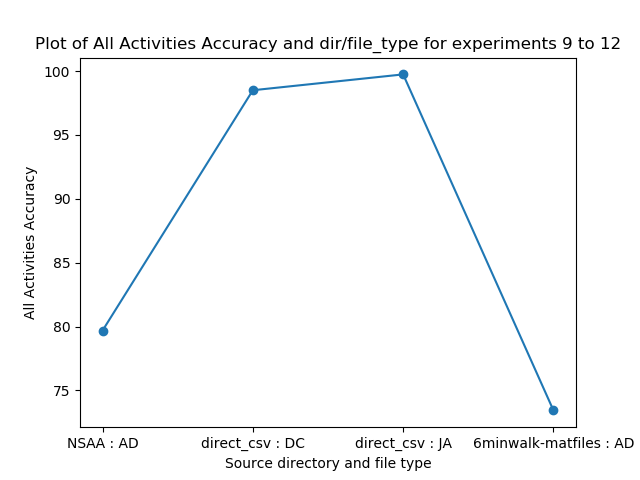
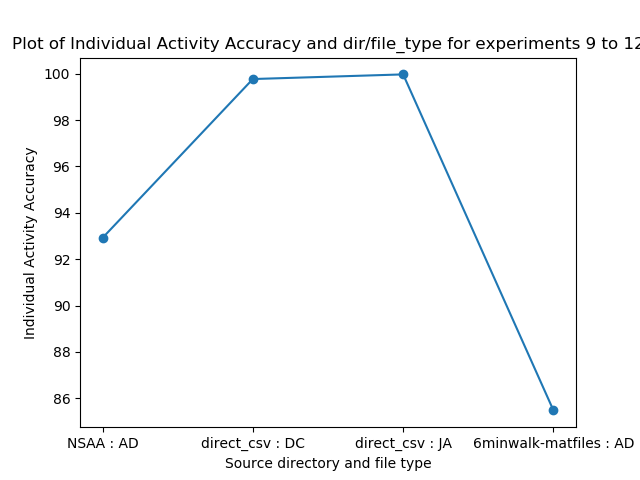
It should also be noted that training the RNN with raw joint angle data requires far fewer training epochs (~20) than for statistical values (~100). This will primarily be due to the larger amount of raw joint angle data that is fed through the network; in the case of training on all files within the data cube, the ‘x’ input shape is (8470, 60, 66) while for the corresponding AD files we only have (552, 10, 30) samples due to how computing statistical values dramatically reduces the raw amount of available data. This decrease in amount of available data also might help to account for the reduction in accuracy when using data sourced from AD files.

Overall, given our limited data and without any way to currently increase it, we come to several conclusions:

1. **RNN models can successfully regress and classify on data 6-minute walk and NSAA assessment data**
2. **Not only this, but raw joint angles perform phenomenally well in both classification and regression**
3. **The raw data measurements might be better put through RNNs than extracted statistical features**
4. **This is probably due to extracted statistical features being a lot smaller of a data set, RNNs better utilizing raw measurements over pre-computed features, and there being a small range of target values with files sourced as ‘JA’ or ‘DC’**

We shall shortly be examining ways to improve the performance of models trained on computed statistical values by increasing sequence overlap, modifying sequence length, and so on.

**Experiment Set 2: Performance of RNNs for Single Activity Classification**



This experiment utilizes the same source data types as experiment set 1. Hence, we won’t discuss what each of these data types represent in the x-axes of the graphs above, as these are exactly the same files as used previously. The difference with this experiment set, however, is that it is looking at a different output type: while experiment set 1 looked at performance for classification of D/HC labels and overall NSAA scores for given test sequences, this experiment set looks at predicting multiple classification values for a single sequence. Here, the models are trained to predict a sequence of 17 values of numbers of either 0, 1, or 2. These represent the single activity scores for a single sequence that correspond to the single activity scores of the file that the sequence is sourced from. It’s worth noting that, as overall NSAA score is the sum of these, a sequence will have an overall NSAA score that is equal to the sum of the same sequence’s 17 single act scores. Furthermore, when predicting single act scores, the RNN architecture will be different: for D/HC classification or overall NSAA score regression, there will be only 1 output neuron (though predicting different ranges of values for the two tasks), while for single acts there are 17 output neurons, 1 for each single act it is predicting. The overall aim of this experiment is thus to see if a model is able to predict, given a sequence of values from a file (that corresponds to a subject, e.g. ‘D4’), what that file’s single-act scores are. Note that the two metrics that we are using (that are computed by each RNN model over all its testing data) is individual activity accuracy and all activities accuracy: the former is the percentage of activities in the testing data set that were correctly predicted to be a 0, 1, or 2, while the latter is the percentage of whole activity sets (i.e. single RNN output of 17 values for a single test sequence) that were correctly predicted.

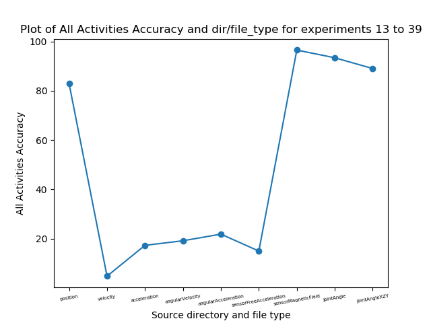
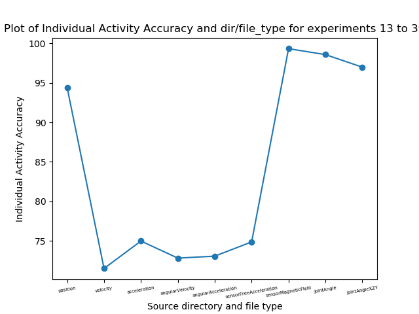
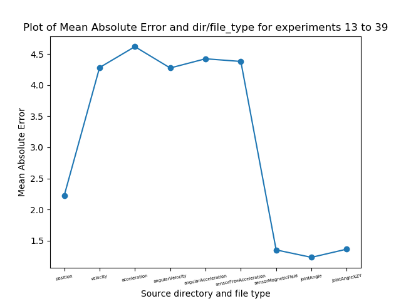
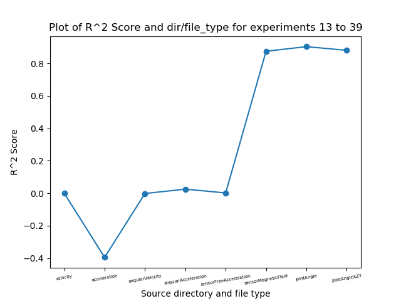
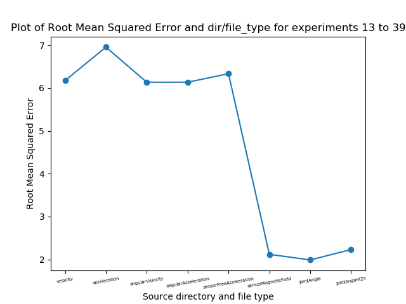
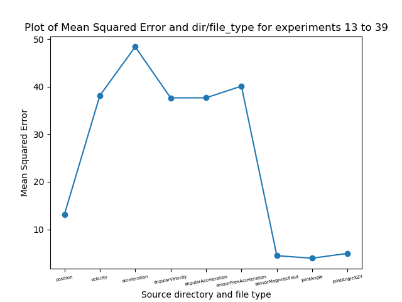
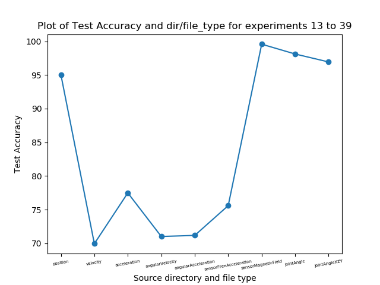
**Results Discussion and Conclusions**

In line with the results from experiment set 1, we can see here that raw joint angle data is much more useful in predicting the single act scores for test data of sequences than stat values from ‘AD’ files. This is most likely down to the same reasons as for the other output types; namely, comparatively small about of ‘AD’ data, neural networks better utilizing raw measurements, etc. Given that it is consistent with the previous experiment set results in determining the comparative performance for different source data types, the results of the experiment set is therefore more about whether it is possible for these data types to train a RNN to predict sequences of values that correspond to activities that the sequence being tested might not be from (i.e. the sequence will be from part of a file that, at most, corresponds to one activity the subject is performing, and therefore can only try to assess what its likely other single activity scores are). The result is that it performs well, especially with joint angle data, predicting approximately **99%** of the individual activity scores and **98%** of the all activity sequence scores.

From this, we can draw several conclusions:

1. **The different types of source data can be used to train a model to accurately predict single act values**
2. **Like the previous two output types, raw joint angle measurements do this better than ‘AD’ stat values**

**Experiment Set 3: Raw Measurements for All Output Types**



With this experiment set, we now turn our attention to just looking exclusively at raw measurements. This is primarily due to the much better performance of models trained on raw measurements compared with ones trained on stat values from ‘AD’ files, as seen in experiment sets 1 and 2. The question we thus ask is: can we show similarly high performance when we train models on types of raw measurements from the suit data other than just joint angles? To this end, we look at 9 raw measurements in total that are contained within the ‘AD’ files and that are recorded by the suit during use: position, velocity, acceleration, angularAcceleration, angularVelocity, sensorFreeAcceleration, sensorMagneticField, jointAngle, and jointAngleXZY. Unlike the joint angle measurements, we don’t have the other raw measurements in separate, unique files. Therefore, we make extensive use of the ‘ext\_raw\_measures.py’ script in order to extract, for every file and for every raw measurement, the measurement data and store them in separate files as ‘.csv’ files. From here, these are then able to train and test an RNN model in the same way.

For this experiment set, we are exclusively concerning ourselves now with NSAA assessment files rather than 6-minute walk files. This is more a choice based on the intended direction of the project to be more concerned with assessing and making predictions concerning the NSAA assessments and how that is connected to natural movement data (more on this later) than the 6-minute walk data. Thus, each of the entries along the x-axis of the graphs represents a single model trained on that particular raw measurement data for every NSAA file we have available. Currently, we are just looking at how the models performs on testing data from subjects it has already seen before, rather than complete subjects being left out of the training set, though this shall be explored later on with ‘left out’ subjects in trained models. Furthermore, though each of the models are trained with the same number of files and each with the same sequence length (due to there at every time step the suit records all raw measurements), the feature size will vary; in other words, if the training shape to the RNN models are of shape (x, y, z), ‘x’ and ‘y’ will always be the same between raw measurements but ‘z’ will vary: for raw measurements based on segment measurements of the suit ‘z’ will be 69, while for angle-based measurements it will be 66 and for sensor readings it will be 51.

**Results Discussion and Conclusions**

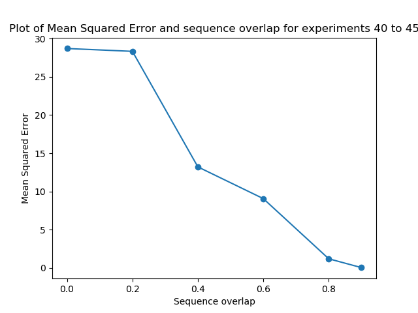
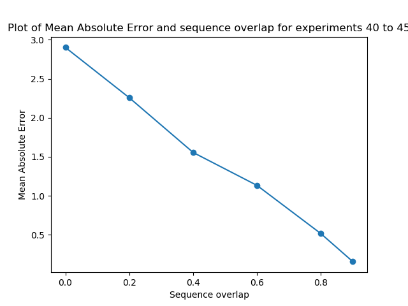
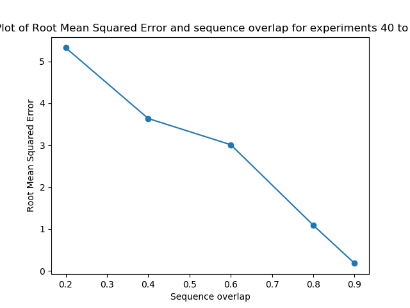
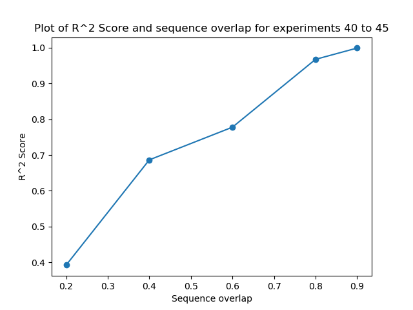
A consistent finding in all of the above graphs is that there are three raw measurements that are far and away better than the others: jointAngle, jointAngleXZY, and sensorMagneticField. These measurements can be seen to perform better for all three output types we are training towards: D/HC classification (graph 1), overall NSAA score (graphs 2 – 5), and individual activities classification(graphs 6 – 7). Both jointAngle and jointAngleXZY raw measurements were to be expected to perform this well, as this is consistent with results obtained in experiment sets 1 and 2 (note that this isn’t using exactly the same files, as experiment sets 1 and 2 use joint angle exclusive files that were pre-extracted before this project’s inception, while the models trained on ‘jointAngle’ here were instead measurements extracted from the ‘AD’ files and represent somewhat different subjects than the ‘JA’ or ‘DC’ files).

Position is another measurement that performs particularly well compared with the other 5 measurements, though for some metrics (e.g. MAE for overall NSAA score) it performs noticeably worse than jointAngle, jointAngleXZY, and sensorMagneticField. We can observe that these 4 ‘useful’ measurements predict **~97%** of test sequences accurately for D/HC classification, compared with only **~73%** for the other 5. Furthermore, when we look at models trained for overall NSAA score regression, we note an average MAE of **~1.5** for the useful 4 measurements, while we see an average MAE of **~4.3** for the other 5; and for single-activity classification for individual activity accuracy, we see **~97%** for the useful 4 and **~74%** for the other 5. With respect to the performance of the useful 4 measurements, while we expect to see performance for jointAngle and jointAngleXZY, the usefulness of sensorMagneticField and position is slightly more surprising. For position, we can speculate this to be as a result of subjects with more severe Duchenne being more inclined to have their limbs and torso in positions which are very indicative of their condition, compared with the healthy-control patients. The sensorMagneticField measurement, however, is slightly more mystifying and does not have an obvious explanation at this point. What’s more, our initial speculation was that the velocity and acceleration measurements would be fairly useful to distinguish between D and HC subjects and, moreover, help predict their overall NSAA scores, as we believed that, as these are key measurements of a subject’s potential for movement, they would be useful for training an RNN model. We found these, however, to perform comparatively badly on test sequences, which leads us to believe that either velocities and acceleration of movement between subjects don’t vary that much or that they aren’t as indicative of movement ability as we thought.

Therefore, the conclusions we draw are as follows:

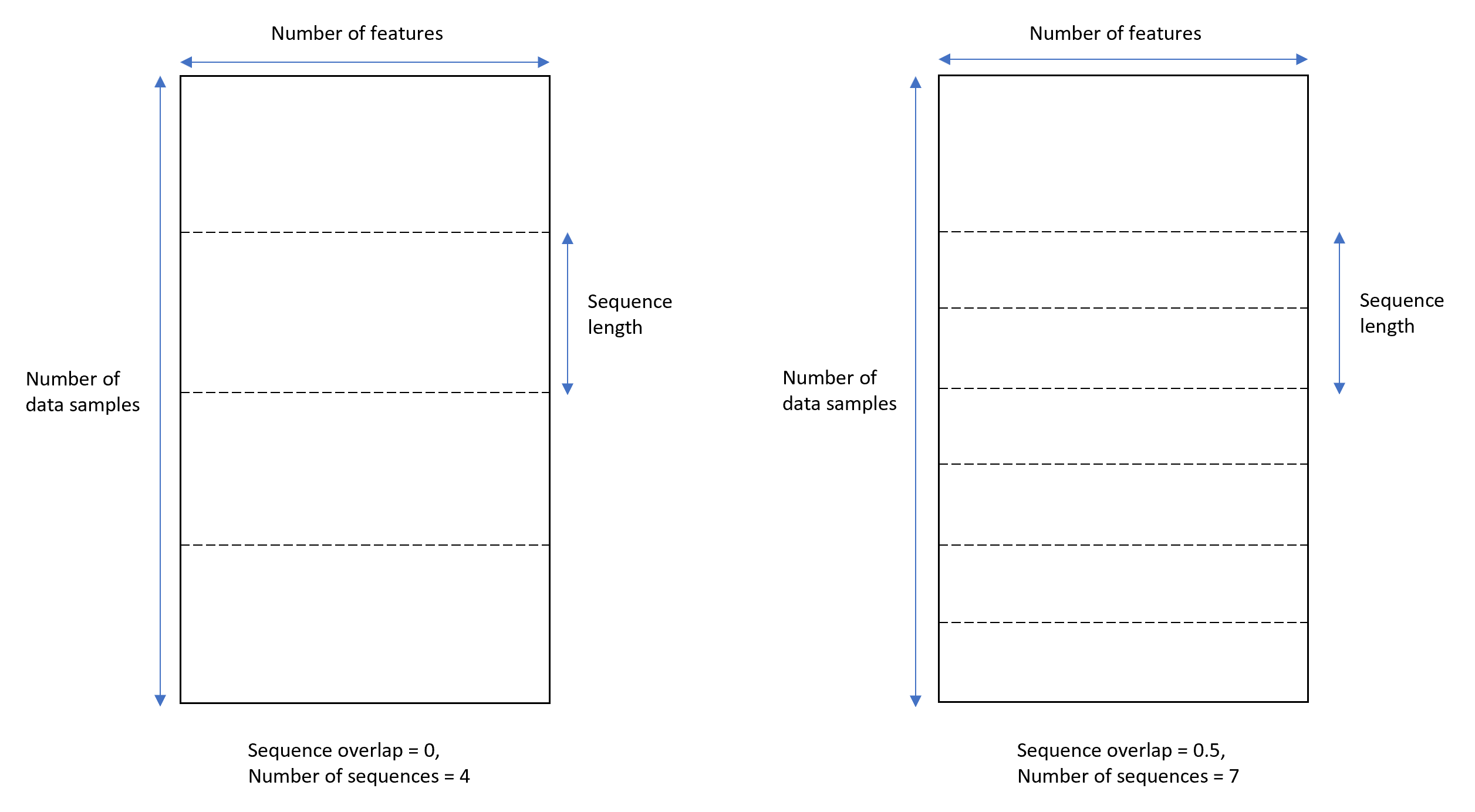
1. **jointAngle, jointAngleXZY, position, and sensorMagneticField are useful raw measurements for building RNN models on, while velocity, acceleration, angularVelocity, angularAcceleration, and sensorFreeeAcceleration are not**
2. **sensorMagneticField is surprisingly well performing (more or less just as good as the joint angle measurements), while position is still strong but slightly worse, and velocity and acceleration aren’t as useful as initially thought**

**Experiment Set 4: Sequence Overlap for Stat Values from AD Files**

One of the problems with stat values extracted from ‘AD’ files via the ‘comp\_stat\_vals.py’ script was that it dramatically reduced the amount of available data we have for training: when we are computing the stat values over intervals of 1 second (the standard measurement used for this project thus far), we are essentially doing calculations over 60 rows of raw measurement data (due to the sampling rate of the suit being 60Hz and we chose 1 second’s worth of data) and producing as an output 1 row of data. While this theoretically contains much of the useful information of the 60 rows simply condensed into 1 row, it doesn’t change the fact that we have now reduced the amount of raw data that we feed into the RNN 60-fold. This is most likely a large factor in the comparatively-weak results of stat values from ‘AD’ files seen in experiment sets 1 and 2. However, a way we chosen to get around this is by using a sequence overlap.

Sequence overlap is essentially used here as follows: consider a 2D block of data that we have available that is produced by the ‘comp\_stat\_vals.py’ script and ‘ft\_sel\_red.py’ to reduce to dimensionality of the data. This block has a number of rows equal to the number of produced rows of stat value data over all files that we are using (e.g. all NSAA files available in the directory) and a number of columns equal to the reduced dimensionality produced by ‘ft\_sel\_red.py’ (e.g. from ~4000 from ‘comp\_stat\_vals.py’ to ~30 to be read in by ‘rnn.py’). Normally, we will take ‘slices’ of this block to produce smaller blocks with a number of rows now equal to ‘sequence length’ before moving on to the next block below it until all data is consumed (ignoring leftover data at the end of the block that won’t fit into a smaller block). This is the case with the diagram below on the left.



However, if we consider a sequence overlap of 50% (i.e. overlap proportion of 0.5), we are instead able to capture 7 blocks of data rather than 4. Scaled up to smaller sequence lengths relative to the number of data samples, this is approximately 50% more data; if we scale up to an overlap of 0.9, we have 900% more data. Though we end up with a fair bit of redundancy with this approach (as the same vector of a data sample is used in numerous sequences), there are two primary benefits of doing this:

1. Much more available data, which is useful to train the models to become much more accurate.
2. With sequences that have no overlap, there is a high chance that one or more activities that occur in the NSAA assessment might be ‘cut’ along these lines (see the dashed lines in the above left image); this would mean the activity isn’t included in an entire sequence, while with a sufficient sequence length and sequence overlap, it’s more likely to be captured in its complete form in at least one of the sequences.

**Results Discussion and Conclusions**

Note that for this experiment set, we are only concerned with the overall NSAA score regression output type, as previous results show the performance of models on this output type are generally very consistent with results on other output types, hence there isn’t expected to be any need to repeat these experiments with the other output types here. Unsurprisingly, we see a massive increase in the performance with a larger sequence overlap. This will be in part due to the massive increase in available data: while the total data for NSAA files from stat values produced by ‘comp\_stat\_vals.py’ for a sequence over of 0 is (742, 10, 30) and shows a **MAE = ~2.9**, when the sequence overlap increases to 0.9, we have total data of shape (7420, 10, 30), ten times as many available sequences, which results in **MAE = ~0.1**. Note that these results are still just on test data, so it shows that the models can generalize much better to new, unseen sequences the overlap is much higher.

We can therefore conclude two things:

1. **A large sequence overlap leads to better performance for stat values from ‘AD’ files for NSAA files.**
2. **This is most likely due to the larger amount of available data and the increased window to capture complete activities.**