**Final Evaluation Results**

**Final Models Directory**

With all the experimentation now complete and the optimal setups of models ascertained (including the different hyperparameters, input types to use, etc.), we wish to extract the models from the set of all created models that we have used for the project. In total, there were 762 models created, many of which were only used for a single experiment set or model predictions set. Therefore, in any effort to separate the models created for experimentation purposes and those being used for ‘production’ (in other words, the models called upon by a user assessing a file in it’s final state), we copied the necessary models from the ‘rnn\_models’ directory located within the local directory and added them to another directory, ‘rnn\_models\_final’, within the project directory under ‘<project directory>\source\’. This was done for several reasons:

* It allows one to use the system completely free of the local directory (thus not requiring the large amount of data to download or models to setup via the data pipeline and the ‘rnn.py’ script). Thus, a user would only need the models within ‘rnn\_models\_final’ to assess files using ‘model\_predictor.py’ and/or a wrapper ‘assess\_’ batch script.
* As the project directory contains the majority of what constitutes the project deliverables, we add in the chosen models as another deliverable in the form of a distinct directory; these are the deliverables as a consequence of many experiment sets and model predictions sets where we looked for the best possible model setups to solve the problems we were looking to solve.
* We wouldn’t necessarily want to include all the models within ‘rnn\_models\_final’, as these will not be accessed via any of the ‘assessor\_’ batch scripts, as by definition they would not be the best models to assess with via ‘model\_predictor.py’ (as they have been proven to be the inferior option via the various experiment sets and model predictions sets); furthermore, they are a form of intermediate data and thus predominantly lie within the scope of the local directory, along with other forms of intermediate data (such as computed statistical values and extracted raw measurements).

With regards to the models themselves, they broadly fall under three categories, which encompass all 48 models contained within ‘rnn\_models\_final’:

1. Models trained on all the subjects (i.e. no left-out subjects) contained within the NSAA data set, with supplementary data from NMB (maximum of 3 files per subject), all of the data available used for training, a sequence length of 600, a sequence overlap of 0.9, a discard proportion of 0.9, and the number of training epochs set to 20. One model trained per combination of input types (joint angle, sensor magnetic field, position, and AD) and output types (‘dhc’, ‘overall’, ‘acts’).
2. Models trained on all the subjects (i.e. no left-out subjects) except for files of ‘V2’ subjects (e.g. a file containing ‘D4V2’ is not used) contained within the NSAA data set, with supplementary data from NMB (maximum of 2 files per subject), all of the data available used for training, a sequence length of 600, a sequence overlap of 0.9, a discard proportion of 0.9, and the number of training epochs set to 20. One model trained per combination of input types (joint angle, sensor magnetic field, position, and AD) and output types (‘dhc’, ‘overall’, ‘acts’).
3. Models trained exclusively on either the NSAA of NMB data set with no subjects left-out, which is then useful in analyzing subject files from an alternative directory. All of the data available is used for training, along with a sequence length of 600, a sequence overlap of 0.9, a discard proportion of 0.9, and the number of training epochs set to 20. One model trained per combination of input types (joint angle, sensor magnetic field, position, and AD) and output types (‘dhc’, ‘overall’, ‘acts’).

It should be noted that categories 2 and 3 will have been created and used by the user undertaking model predictions sets 21 and 23, respectively, while models from category 1 will have been created independently by running the ‘models\_no\_leftout’ script found in ‘<project directory>\source\batch\_files’. This is because, while we have very similar models that will have been created in MPS 20, these all contain one subject left-out of the training set (so as to assess this subject on the models to test generalization ability). If we wish to test subject files in production, we want to use models that have been trained on all available training data; hence, the ‘models\_no\_leftout’ is run to create these models, which constitute category 1.

**Utilizing the Final Models to Assess New NSAA/NMB Files**

With the models chosen as the best performing models to generalize towards new files and that have been trained on all available NSAA data we have (along with being supplemented by NMB data), we were now in a position to create a final Python script that is properly able to utilize these models. While ‘model\_predictor.py’ was written in order to assess a variety of files on different models with different arguments set, the ‘assess\_nsaa\_nmb\_files.py’ script was written to exclusively operate using only several of the models built in total (specifically some of those in ‘rnn\_models\_final’). The script itself calls ‘model\_predictor.py’ itself with the setting of certain arguments based on user input. Additionally, it also does a lot of the data preparation of the data that is to be fed through ‘model\_predictor.py’. In other words, ‘assess\_nsaa\_nmb\_file.py’ is a wrapper that sits over all the other scripts that, based on user supplied argument at runtime (as opposed to through arguments), calls the appropriate parts of the data pipeline and the models to use from ‘rnn\_models\_final’ to assess the file. For further information about the script and an exact breakdown of how it works, see the ‘assess\_nsaa\_nmb\_file.py’ script within the ‘Script Ecosystem Overview’ chapter.

While the majority of the Python and associated batch scripts would be used by other users intending to either replicate results outlined in this report or to continue the work already done by the project, the ‘assess\_nsaa\_nmb\_file.py’ script has a slightly different focus in that it’s intended to be able to used as a tool in a production setting. Below, we consider several use cases of the script where the project directory has been obtained, the Python language (and requisite packages have been installed to run the script), and the user has one or more source ‘.mat’ files they wish to analyze:

* Given that the ‘.mat’ file the data (either of natural movement or NSAA assessment) of a brand-new subject to the research initiative, one could use the script to give the system’s opinions regarding the subject’s overall and individual NSAA scores (the D/HC will most likely already be known to the user)
  + In the case where it is of an NMB data file, it would allow the user to possibly avoid the requirement to do an NSAA assessment for the subject if they aren’t able or willing to do so but can instead provide them with natural movement data using the suit.
  + In the case where it is an NSAA file, it would be able to be assess the subject who’s file it corresponds to and give a supplementary opinion regarding their scores (e.g. if the assessor is unsure about certain scores or for whatever reason has not provided them).
* Given a new ‘.mat’ file of a new ‘version’ of an existing subject (e.g. the suit data of a subject’s 2nd assessment, 6-months apart from their previous one, at the hospital), it could provide an easy way for the assessor to see how much worse the subject has gotten between their true previous scores (i.e. for their ‘V1’ visit) and what the model predicts as their scores from their new ‘V2’ visit.
* If the user was unsure about the results they achieved either from the system or from their own assessment, they could opt to test the ‘.mat’ file on models trained on an alternative directory to that of the source ‘.mat’ file so as to receive a different opinion from a different set of models.