**Further Improvements and Additional Project Adaptations**

While an extensive amount of refinements and improvements have been undertaken during the course of this project, there still remains a great deal of improvements that can be made to the system and different ways we can adapt the system to either different project domains or to the same domain but with using different technologies. Many of these involve tasks and modifications not made due to time constraints or modifications that exist outside the scope of the project but that could be added if the scope was slightly modified. In the sections below, we shall outline a few of each of these with the hope that one could use these as a starting point to continue with the project to either improve performance, adapt it to conditions other than DMD, work with different types of source files, and so on.

The immediate improvements that could be made include the following:

**Further modifications of RNN hyperparameters**

* While there was an extensive amount of system modification tested through experiment sets and model predictions sets to ascertain the optimal combination of measurement data to be used, the best sequence hyperparameters (sequence overlap, discard proportion, etc.), there were many hyperparameters that simply untested and we ended up using the same values throughout the project.
* This includes the number of hidden layers used in the models (2), the learning rate of the optimizer (0.001), the algorithm used as the optimizer (‘adam’), among other hyperparameters. The values chosen for these were generally chosen as they were either good ‘rules of thumb’ with respect to the hyperparameter (e.g. the learning rate of 0.001) or we have previously seen good results in other projects using the setting (e.g. in using the ‘adam’ optimizer).
* These, of course, are no guarantee of ideal performance for the problems we are trying to solve in this project, and the primarily reason why alternative values for these hyperparameters weren’t explored is due to time limitations. Furthermore, it was felt that exploring other system settings in our experiments would give settings that also would give an insight about the data itself; for example, finding out during our experiments that the joint angle measurement was much more useful for building models with than the velocity or acceleration measurements leads us to more insightful conclusions about movement data than finding out that a learning rate of 0.001 is more optimal than either 0.01 or 0.0001.

**Use of greater compute resources to run the data pipeline and build models faster**

* While the local PC used for the majority of the work of the project has proved adequate for our purposes, it also showed limitations in some of the more demanding tasks. For example, to run the whole ‘setup.cmd’ script that creates all the necessary intermediate data that is used to create the models, running the script with a high-end processor (e.g. ‘Intel i7-8750H’) still requires upwards of 10 hours to fully complete. Additionally, for each model created with a moderate-to-high-end GPU (e.g. ‘GeForce GTX 1050ti’), for the typically-sized data set used to train a model it takes between 5-10 minutes to create: in the cases of model prediction sets were we need up to 60 models to be created, many hours might be needed per MPS.
* An alternative to this would be to make use of the Condor computing framework made available by Imperial College London’s Department of Computing in order to submit these as batch jobs. This would enable to user to complete the setup and model construction phases of the project fairly quickly, which should make collaboration future work on the project more feasible for users who do not have many hours available to do the necessary setup and model construction.

**Exploration of alternatives to sequence modelling than RNNs**

* For this project, we have focused solely on using RNNs with LSTM units as our sequence modeller of choice with which to build models. However, there are several alternatives that were considered but were not explored to keep the project scope focused on a single adaptation to a solve a specific problem.
* These include Hidden Markov Models and Conditional Random Fields, which can both be implemented within Python and could theoretically work with the sequences that we create in the system to predict either classification or regression targets.
* Therefore, a feasible next step in the project’s lifecycle could be the replacement of the ‘rnn.py’ script with a script that builds from the same data a conditional random field to predict on the same output types as ‘rnn.py’. From this, one might find out that a different class of sequence modelling produces better models for the data sets what have been using than RNNs

Additional adaptations of the project work include the following:

**Working with source ‘.mat’ files of subjects with different movement conditions other than DMD**

* We could very easily modify the system to work with data of subjects with a different type of condition other than DMD; this could possibly be another form of muscular dystrophy or any other condition where subjects’ movements different from what would be considered ‘healthy’ movement.
* If this condition is also captured from the subjects by the Xsens body suit and if the data is then converted from ‘.mvnx’ to ‘.mat’ format, it will contain the data in the same structure as we are used to; hence it will be processed the same way. Alternatively, any movement capturing software that produces ‘.mat’ files with data in the same format as the source ‘.mat’ files used here would be acceptable for the data pipeline.
* The only other necessary information needed for the system would be an annotation file in the same format as ‘nsaa\_6mw\_info.xlsx’ to contain the requisite labels for various subjects. Additional modification of the ‘preprocessing()’ functions used by ‘rnn.py’ and ‘model\_predictor.py’ scripts would be necessary to build models that train towards these different ‘y’ target labels. For example, if the new condition (other than DMD) that we have source ‘.mat’ files on and that are used to train models needs to be classified as being of several classes (i.e. an extension of binary classification as we used for the ‘dhc’ output type), the ‘preprocessing()’ functions would need to be modified to reflect this.
* With the new condition data obtained (with the data sets in the local directory and modifying the ‘sub\_dirs’ variable in ‘settings.py’ to point to this), the annotation file setup as a replacement to ‘nsaa\_6mw\_info.xlsx’ (modifying the ‘nsaa\_6mw\_path’ variable in ‘settings.py’ modified to reflect this change), and the necessary preprocessing functions modified, we could feasibly build models on this data which we can then use to assess new subjects with the condition in the same way as we have done with the DMD subjects.
* In this sense, the framework for the system would be nearly identical, with only minor modifications needed to adapt the system to work with this new data. Therefore, it’s fairly trivial to set this up as continuation work to get the system to work with new forms of data.

**Working with sequence-based biomedical data of non-‘.mat’ data sets**

* Taking the project at its highest level, we essentially have a system that takes in source ‘.mat’ files, extracts sequences from them either from raw or feature extracted measurements, trains models on this data, and assesses new files using these models.
* While the above section provides an outline for data for different conditions that exists in a similar or identical format as what we are used to, we could feasibly heavily adapt the system to work with different sorts of source data files (e.g. ‘.csv’ files) and of non-human-movement based conditions. In this case, the complete system created by this project would serve as a ‘roadmap’ for one way to build a complete data preprocessing, model training, and file assessment system, which has proved to be a success for this project. Heavy modifications, however, would have to be made to the scripts outlined here in order for them to work with this drastically-different data set.
* The only real requirements would be that the data can be loaded and manipulated using Python, the data can be sensibly divided into sequences (e.g. not image data, as the images themselves are independent of subsequent images in most cases), and we have information that allows us to create true labels of the data we are given (i.e. an equivalent to ‘nsaa\_6mw\_info.xlsx’ to provide classification labels or regression values that correspond with the created sequences). In this sense, we could use the skeleton of the system outlined in this project to provide the basis of a completely-new project (e.g. sequence modelling of medical records to predict patient risk of various medical conditions).