The input taking screen looks like the first image and allows a set of space separated values for each question. After typing parameter sets, it submits the jobs. The results are saved in our group scratch on four folders (detailed\_results, job\_files, runset\_summaries, waiting\_jobs). A live summary of the run sets are continously updated and allows to view resulting objective values of all the runs together (see image 2). The detailed results of individual runs (log files, images, or what you specify in write\_actions) are saved in detailed\_results. Also, the error files and waiting jobs are written. You can use Sherlock's ondemand web interface for looking at these results.

To use it, you need to modify according to your project some lines on

* the parameters dictionary (params\_dictionary) (see image 3. it has kind of a syntax and allows you to specify the sets of parameters. it has a bunch of features (which i do want to bother you with telling) for input validating, specifiying parameter's default, conditionally requiring some of parameters, putting conditions etc.)
* job script creator python file (runset\_train/submit\_job\_files\_given\_opts.py) (this contains job script creation. you may want to request certain amount of resources, load some more modules, change job time, type your email address etc.)
* of course the models folder (model\_xxx.py specifies the model arch., write\_actions\_xxx.py specifies what to write to logs or what to save in detailed results, plot\_xxx.py file is optional and specifies what to plot)

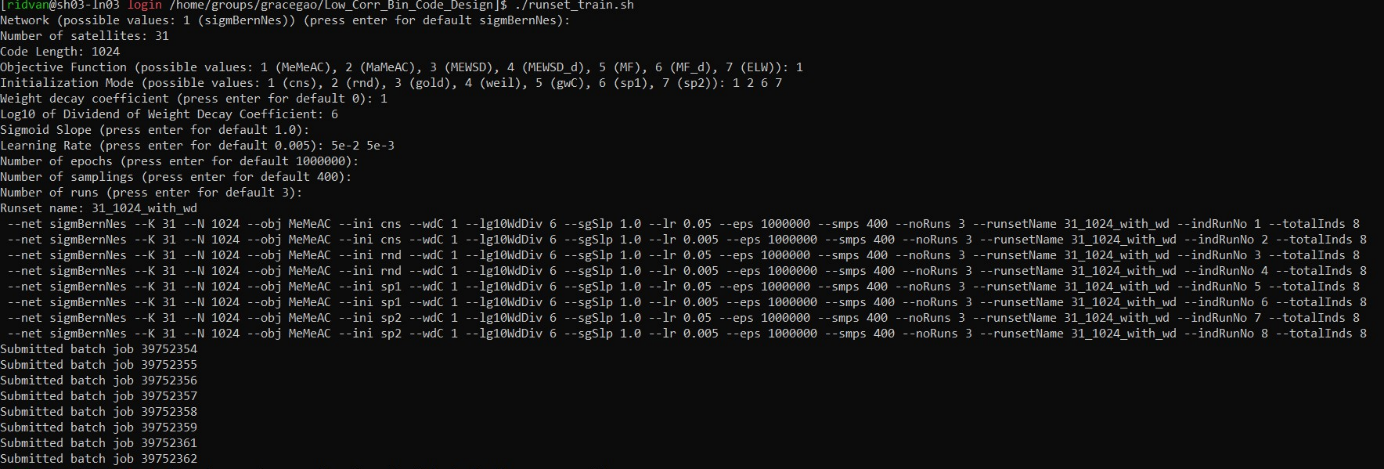


Image 1: The input taking screen

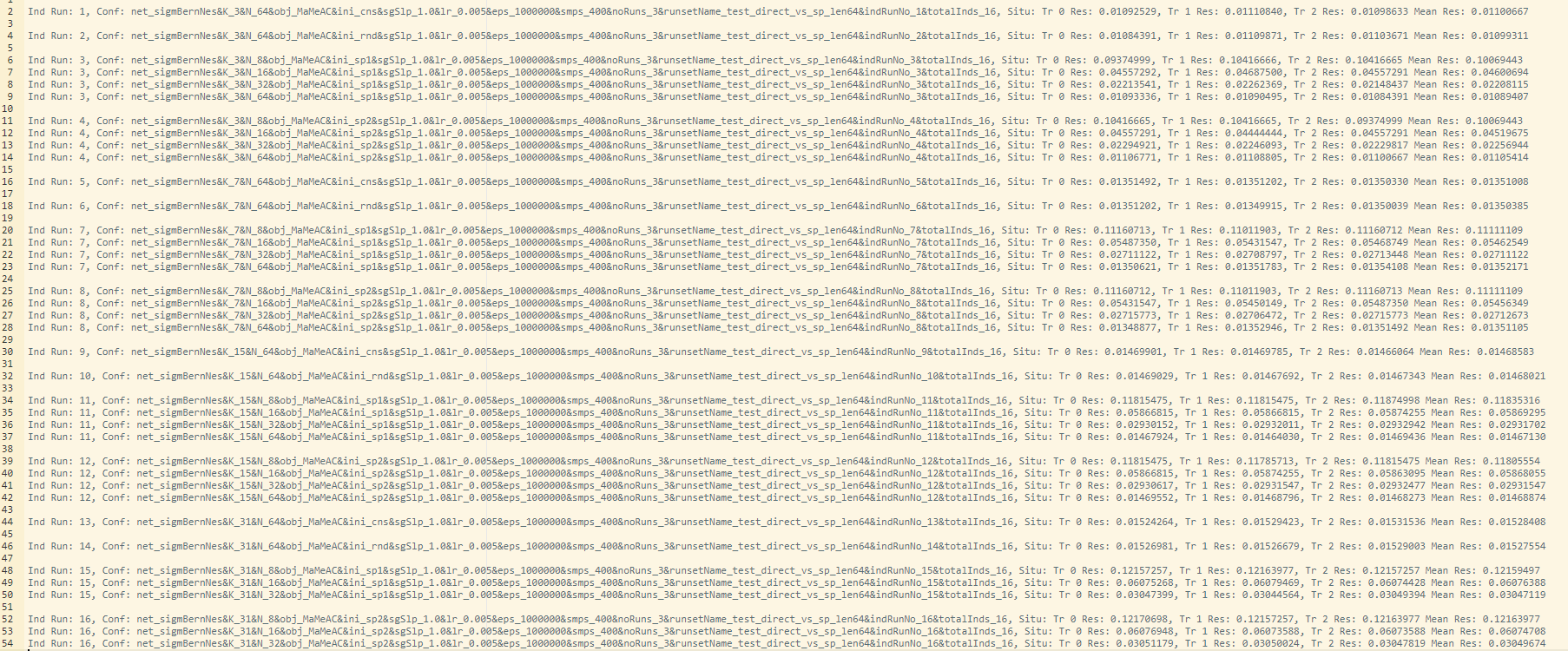


Image 2: An example live summary of results of a set of trainings (16 trainings)

A picture containing text

Description automatically generated

Image 3: An example of the parameters dictionary