User Manual: Using RPM with GrammarViz

May 26, 2017

Contents

1 The Data

1.1 Types of Data

Training Data Training data is the primary data and will be used to create a model that can identify similar patterns in new, unlabeled, data. This data must have a label for each time series so that RPM can learn what the labels can look like. This is where the bulk of the data shall be as RPM will need many samples to find representative patterns.

Testing Data Testing data is a small subset of data usually from the same source as the training data but not found in the training data. This set of data will be used to test the model that RPM made for accuracy.

1.2 Data Format

Figure 1: Examples of RPM Data

```
# 1 1 1 2 2 2 2 2 2
0 0 0 122880 122880 122880 122880 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 122880 122880 0 0 0
```

(a) Example 1

```
# 1.0000000e+000 1.0000000e+000 1.0000000e+000
-4.6427649e-001 -8.9697208e-001 -4.6469596e-001
-5.5504787e-001 -6.8568553e-001 -5.6773891e-001
-8.4284310e-001 -1.3513818e+000 -3.2022764e-002
-8.6589548e-001 -1.4586668e+000 -6.3504562e-001
-9.3639631e-001 -1.1653456e+000 -6.0282554e-001
-8.1726995e-001 -1.4039293e+000 -2.6685628e-001
-2.6361216e-001 -1.8217996e+000 -2.6706128e-001
-1.2580483e+000 -8.3160109e-001 -9.3104230e-001
-1.2503934e+000 -1.0163124e+000 -4.4938186e-001
-9.1830825e-001 -8.0353040e-001 -7.2134200e-001
-9.2210226e-001 -1.2595048e+000 -3.9727192e-001
-9.8448828e-001 -1.1392341e+000 -9.6212589e-001
-1.2880511e+000 -8.7865203e-001 -1.4206669e+000
```

(b) Example 2

RPM Data The formatting of the data is important because the system will not run unless it can read that data. The data files are simple text files that store the time series data with one entry per column, with a space delimiter, with each row representing a time step in the time series data. With RPM compatible data the first row in the file is prepended by a "#" with rest of the row containing the label for each entry rather then the time series values. Keep note that GrammarViz data has the same format except it does not have the row containing the labels, if the data is missing this row RPM will not be enabled in GrammarViz. Also note that labels are treated as strings so there is no requirement that they be numbers. Examples of RPM compatible data can be seen in figure 1.

2 Using RPM

2.1 Training RPM

Once you have the data in the proper format and GrammarViz open training RPM can begin.

Step 1 First click on the "Browse" button under the "Data Sources" section of the window, as seen in figure 2.

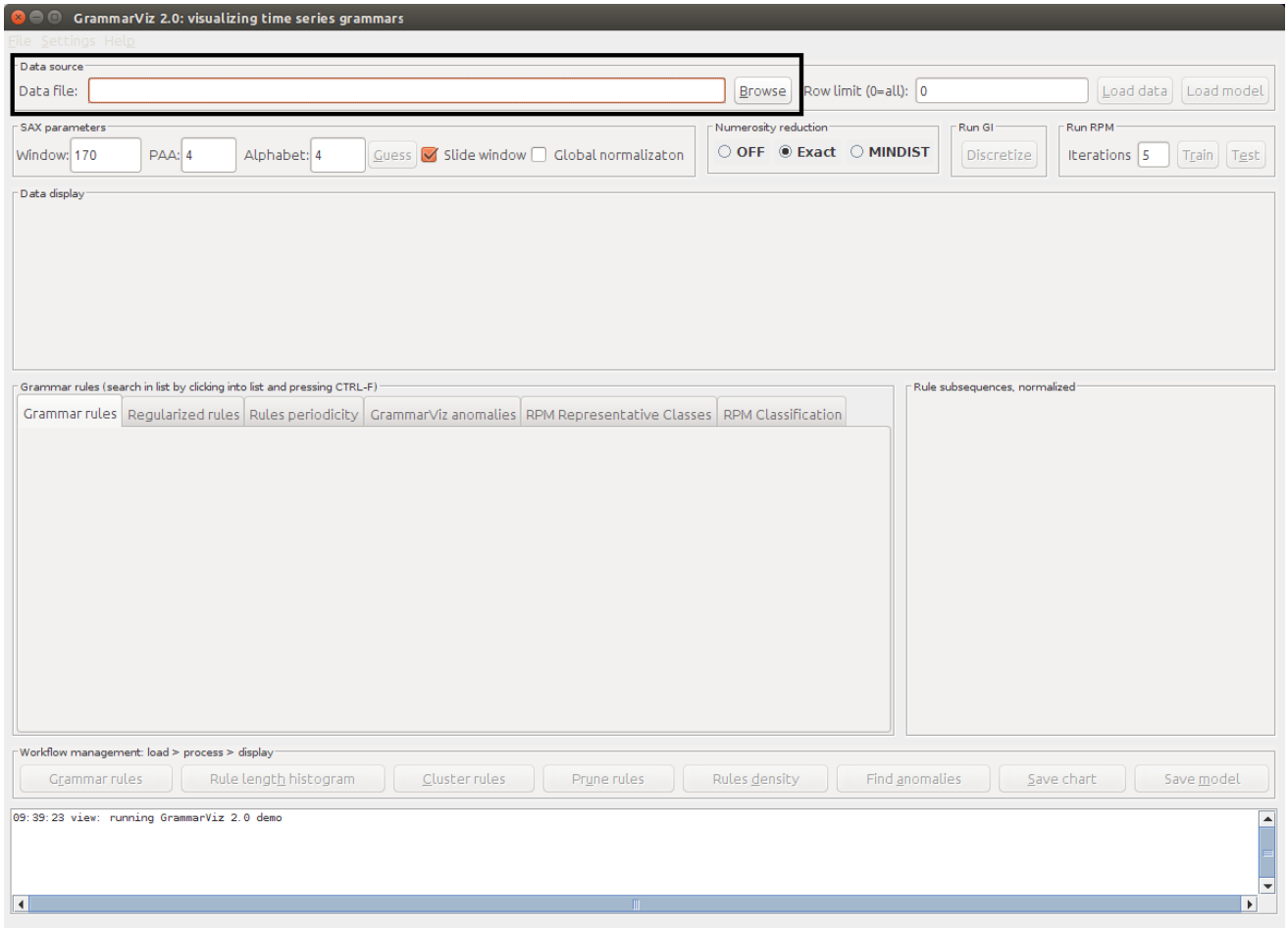

Figure 2: Open GrammarViz

Step 2 This should being up the file browser prompt in figure 3. Using this prompt select the file containing the training set in the RPM compatible format, figure 4.

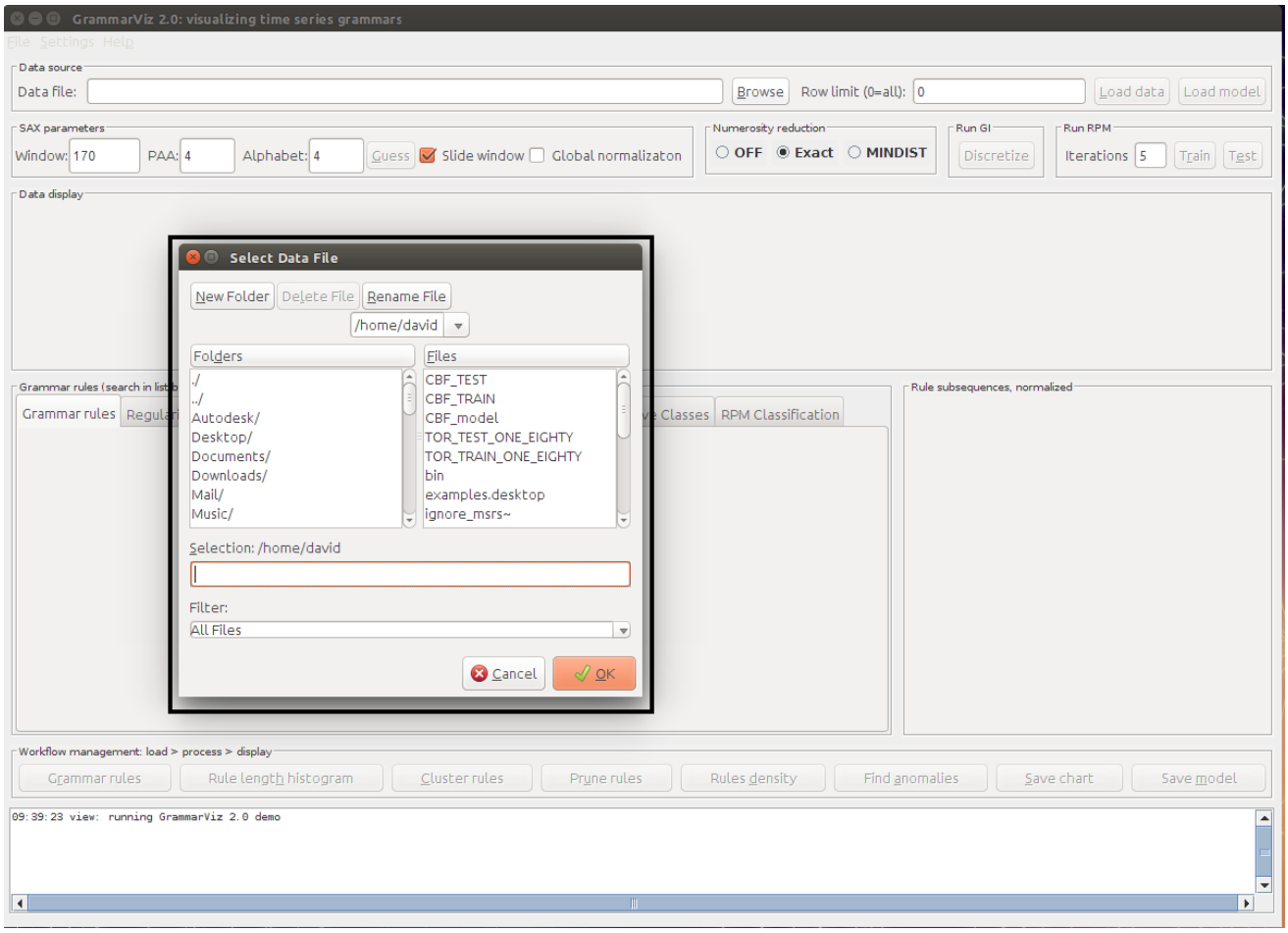

Figure 3: Open the file browser prompt

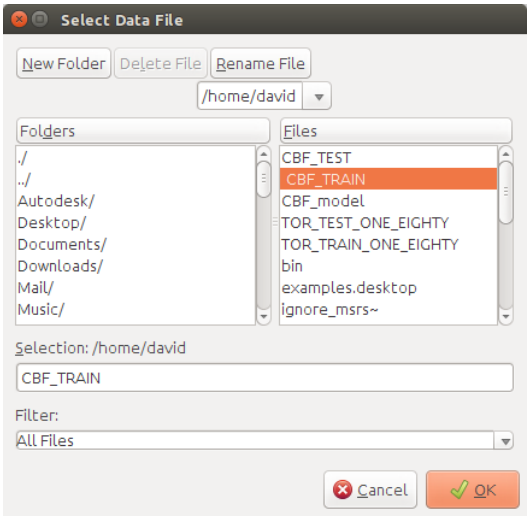

Figure 4: Browser prompt

Step 3 After selecting the file press the button labeled "Load Data" and GrammarViz will load the data and the graphs will be populated, and if the data is found to be RPM compatible data then the "Train" button should become available. The text field labeled "Row Limit" allows the user to limit the number of rows that are read in from file, for example if the file contains 100 rows the user could limit it to the first 50.

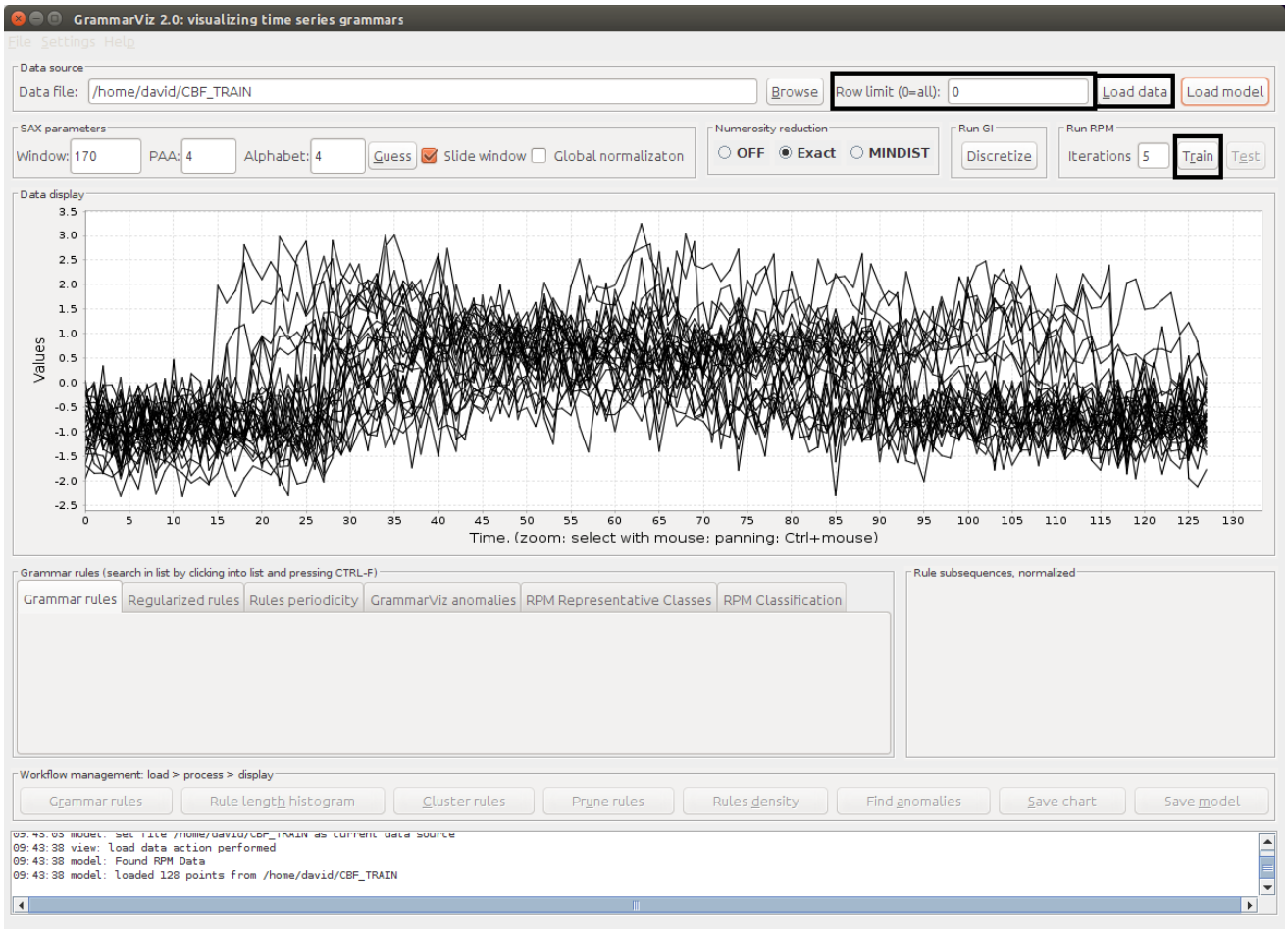

Figure 5: Loaded data

Hitting this button will begin the training phase of RPM, this can take some time depending on the data and the number of iterations RPM will run. The text field labeled "Iterations" sets the maximum number of iterations RPM will go, this prevents RPM from running for to long trying to refine the model. Once the training is complete the tab "RPM Representative Classes" will become populated with patterns RPM thinks be represent the labels given. The fields "Window", "PAA", and "Alphabet" will also be populated with the values RPM believes are the best fit for the data to aid in further analysis.

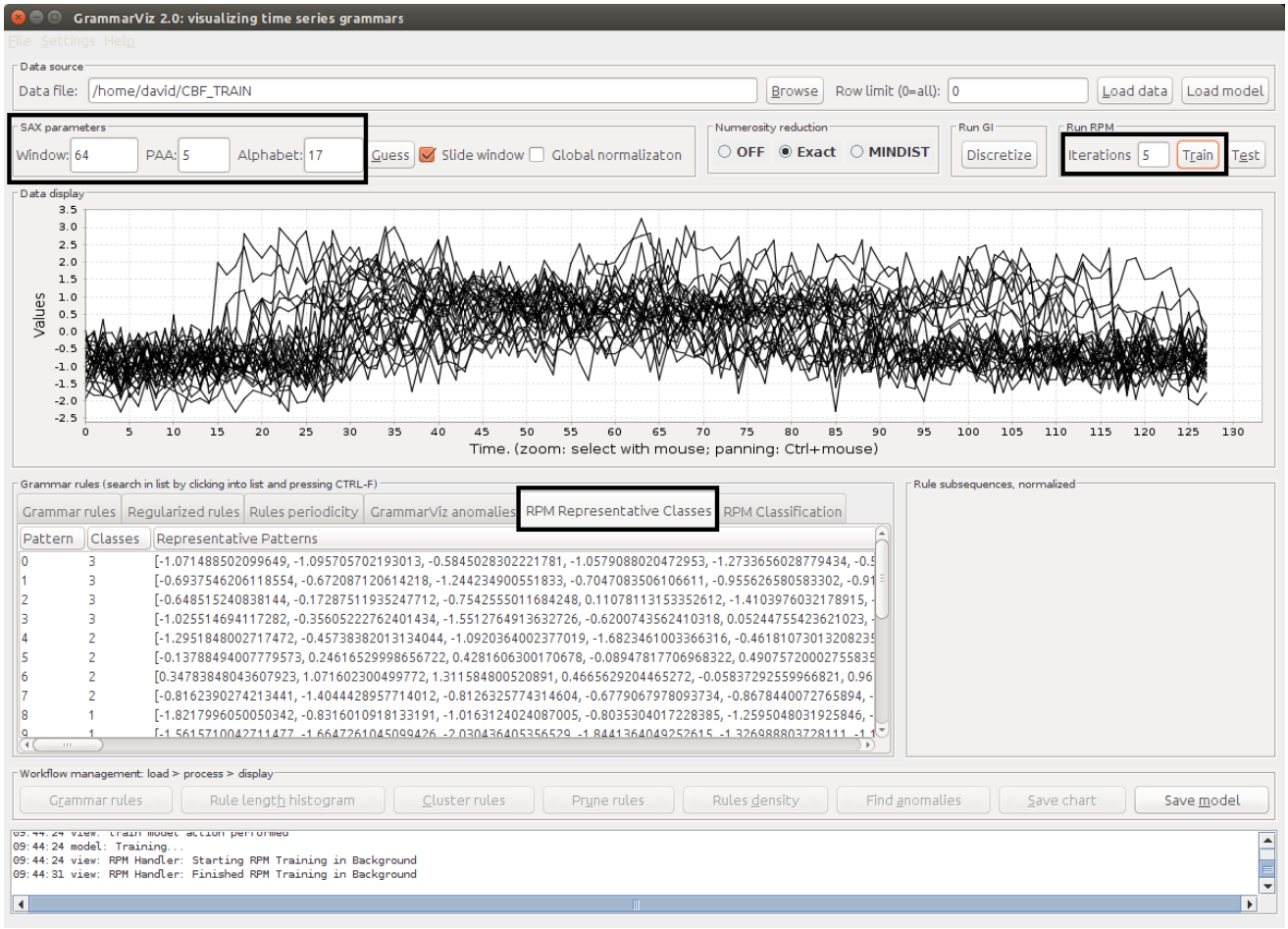

Figure 6: Representative Classes after Training

Selecting the patterns will display their graph on the right hand side of the window, multiple patterns can be selected.

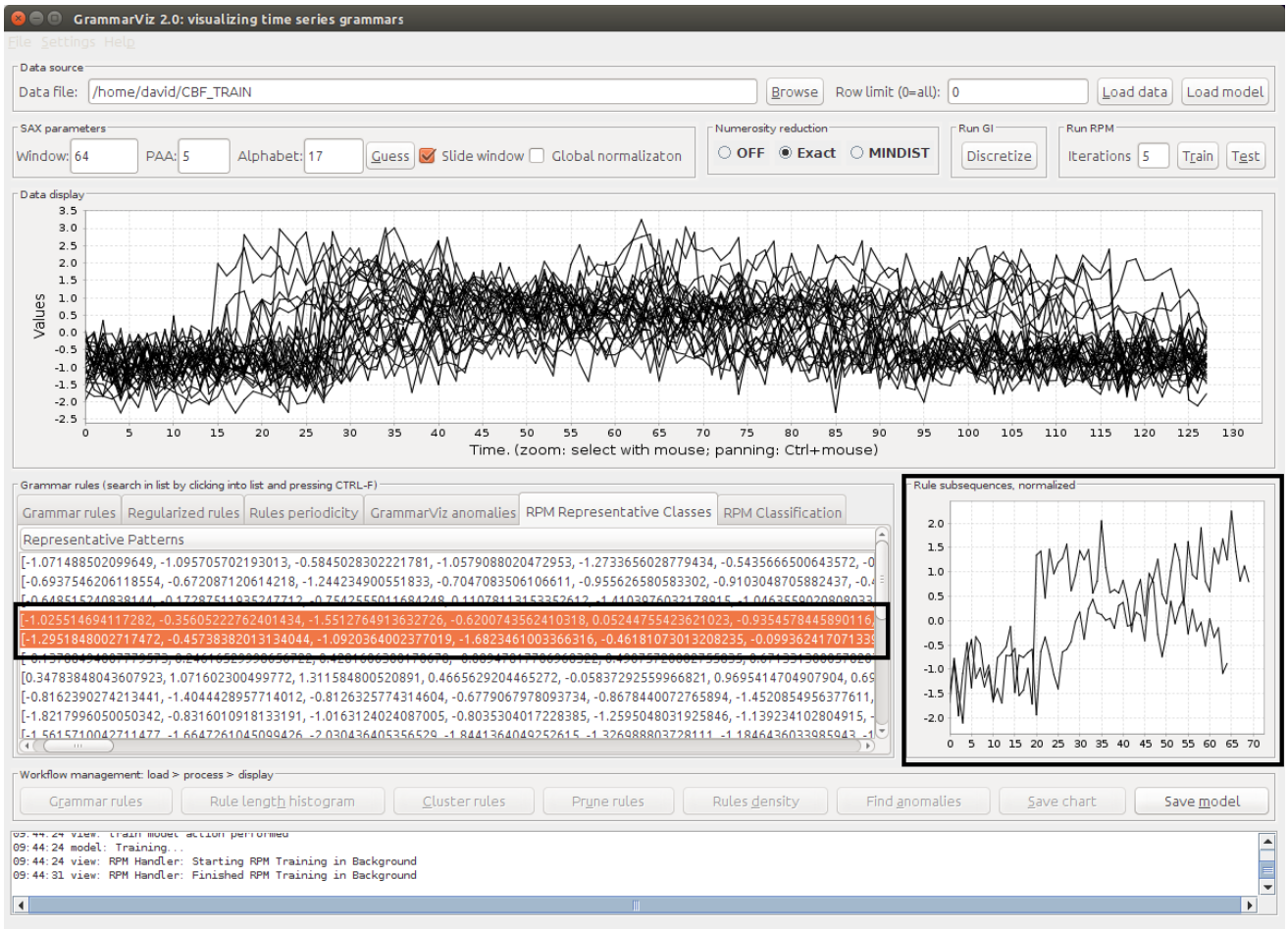

Figure 7: Representative pattern preview

2.2 Testing the RPM Model

Once the model has be trained it should be tested for accuracy, this will use a smaller dataset in the RPM compatible format to measure how well the model does.

Step 1 Click the "Test" button and a file browser prompt will appear, depending on how large the dataset is this may take a moment.

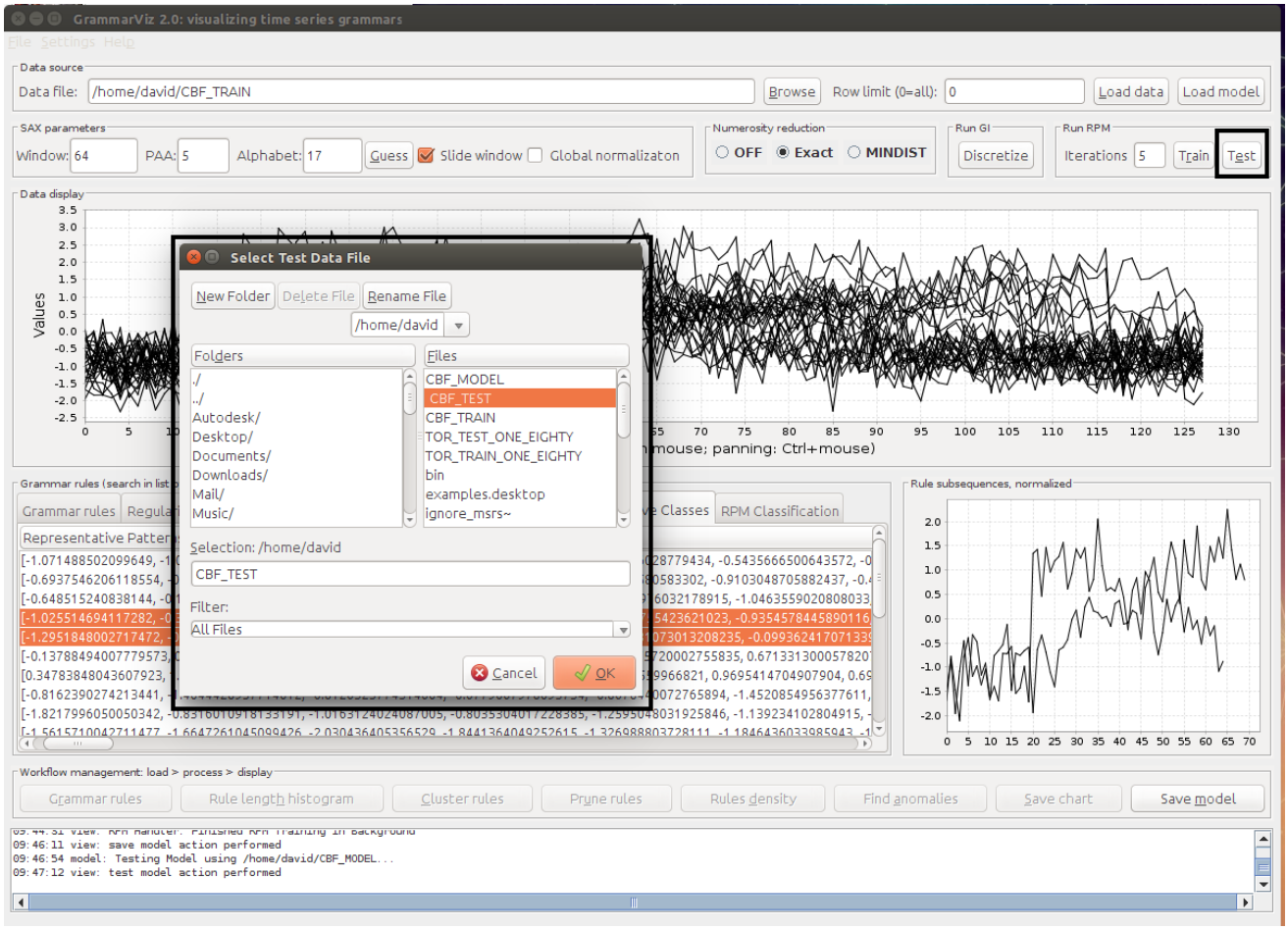

Figure 8: Testing the RPM model

Once the testing is complete the tab labeled “RPM Classification” will be populated. This provides statistics on the effectiveness of the model by reporting the number of samples that were incorrectly labeled by the model.

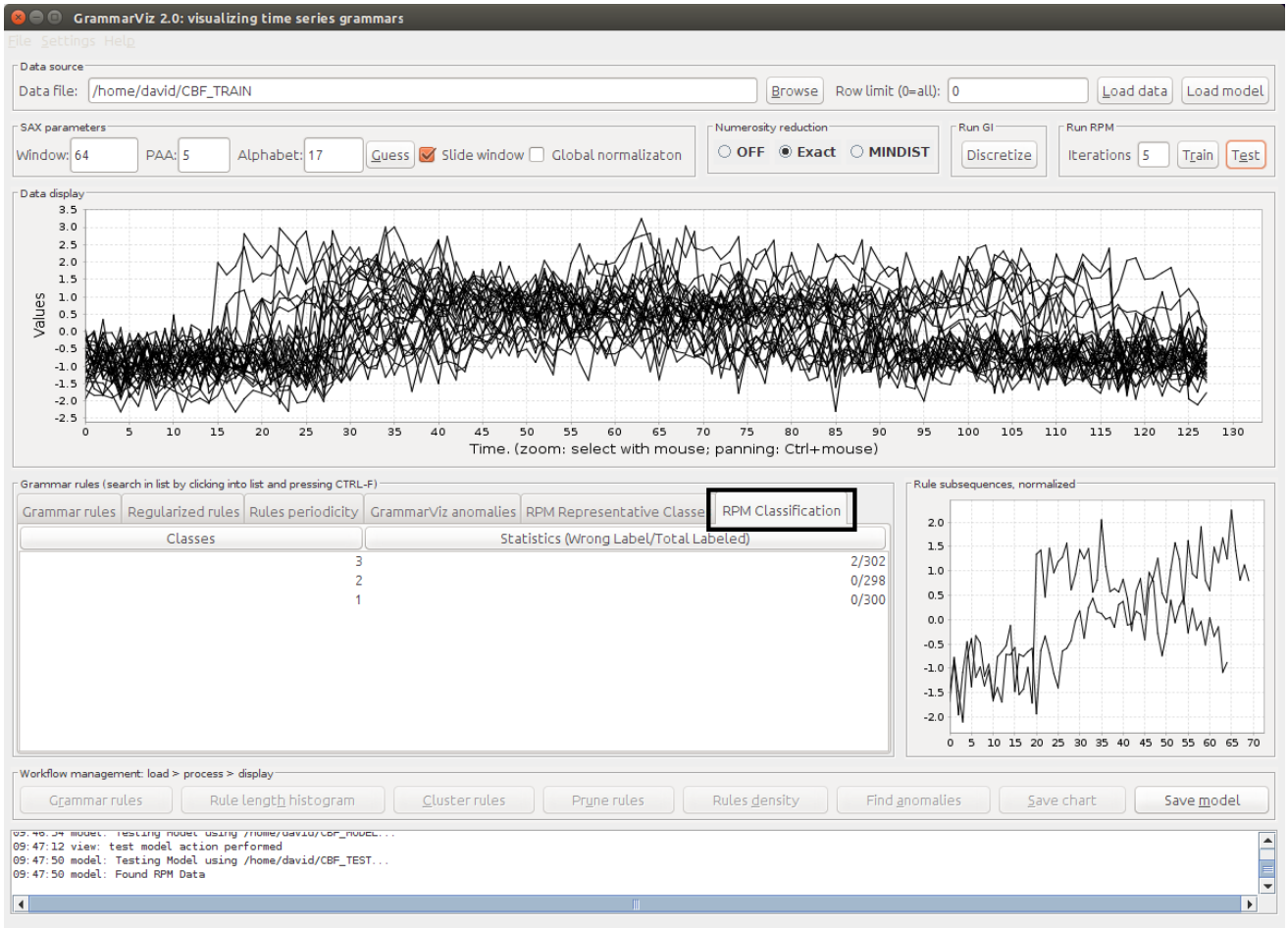

Figure 9: The results from the testing

2.3 Saving an RPM Model

Creating a model can take some time and there for being able to save the model for later uses is a useful feature. Saving the RPM model will generate a file that can be loaded in later for further testing. One thing to note is that the saved model does not contain the training data however the training data is still needed when doing testing there for a copy of the training data must be retained.

Step 1 Once a model has been trained up clicking the save model button, as in figure 10, a file browser prompt will appear.

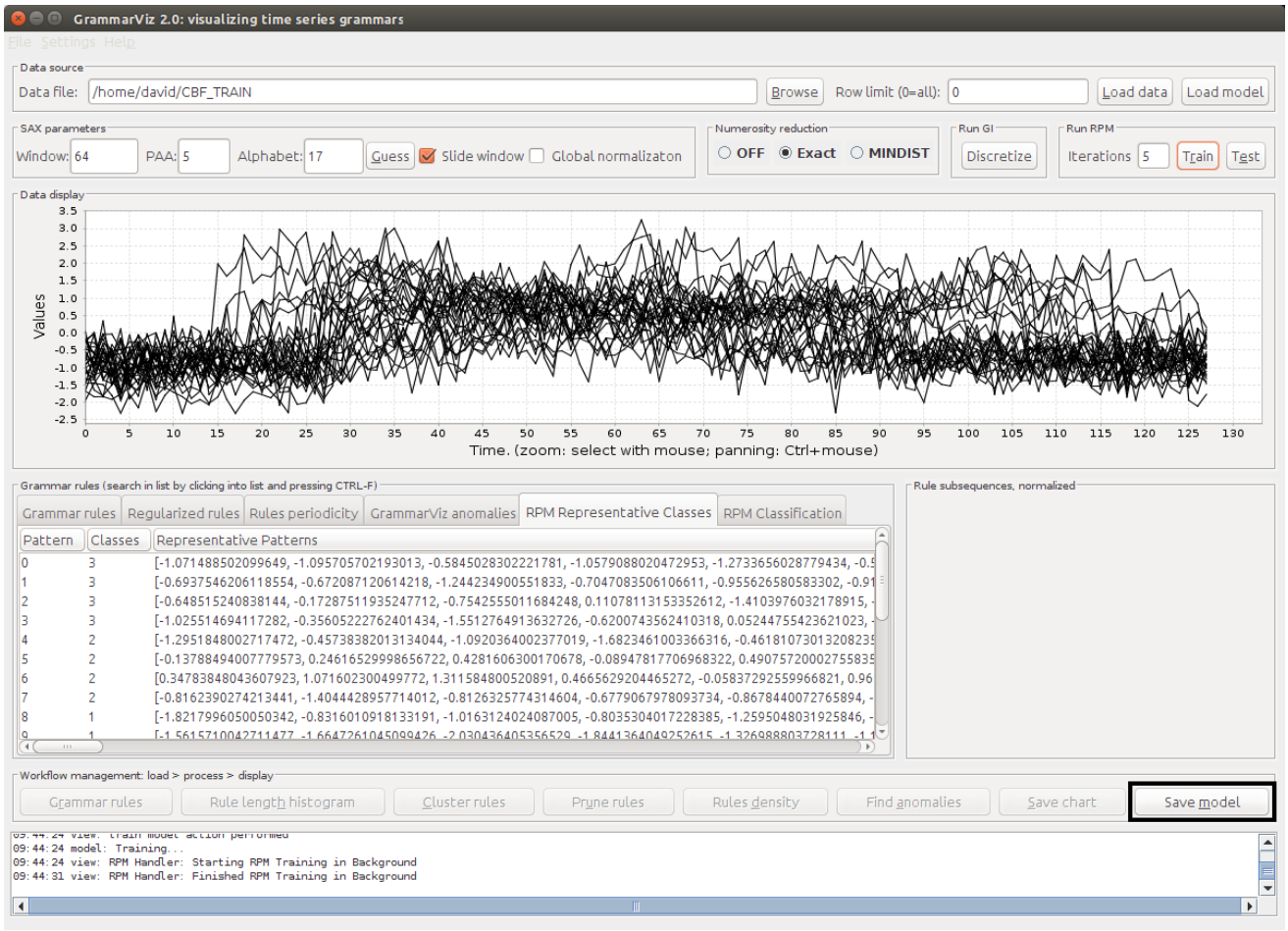

Figure 10: Saving the RPM model

Step 2 With the file browser prompt select a location to save the model and give it a name, then click the "OK" button to save the model.

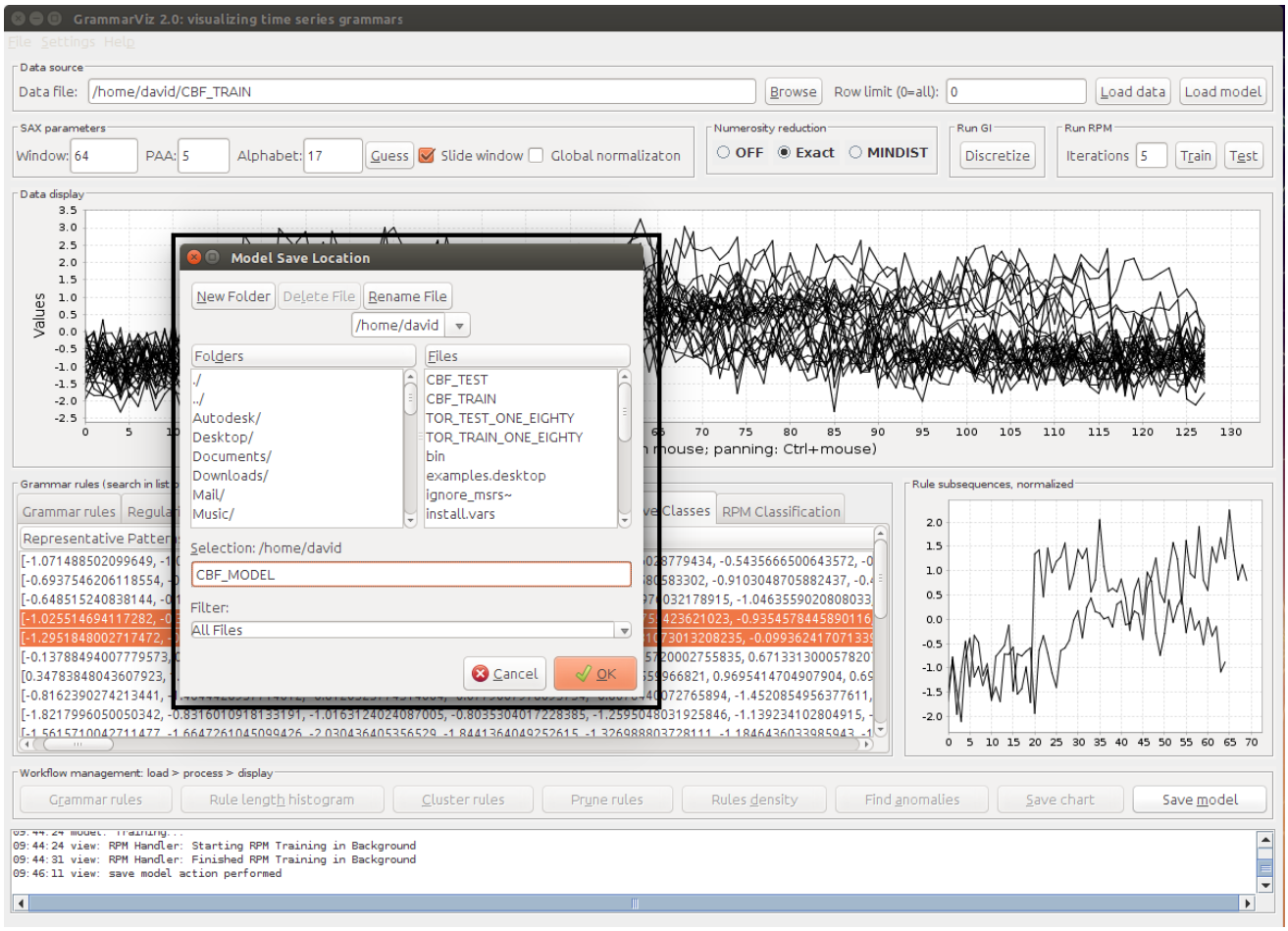

Figure 11: Saving the RPM model to file

2.4 Loading an RPM Model

When a model has already been saved, simply loading the will allow for further testing. When loading a model the software will look for the original training data from where it was when it was originally trained. If the data is not there then the software will ask for the location of the data.

Step 1 First click on the "Browse" button under the "Data Sources" section of the window, as seen in figure 12.

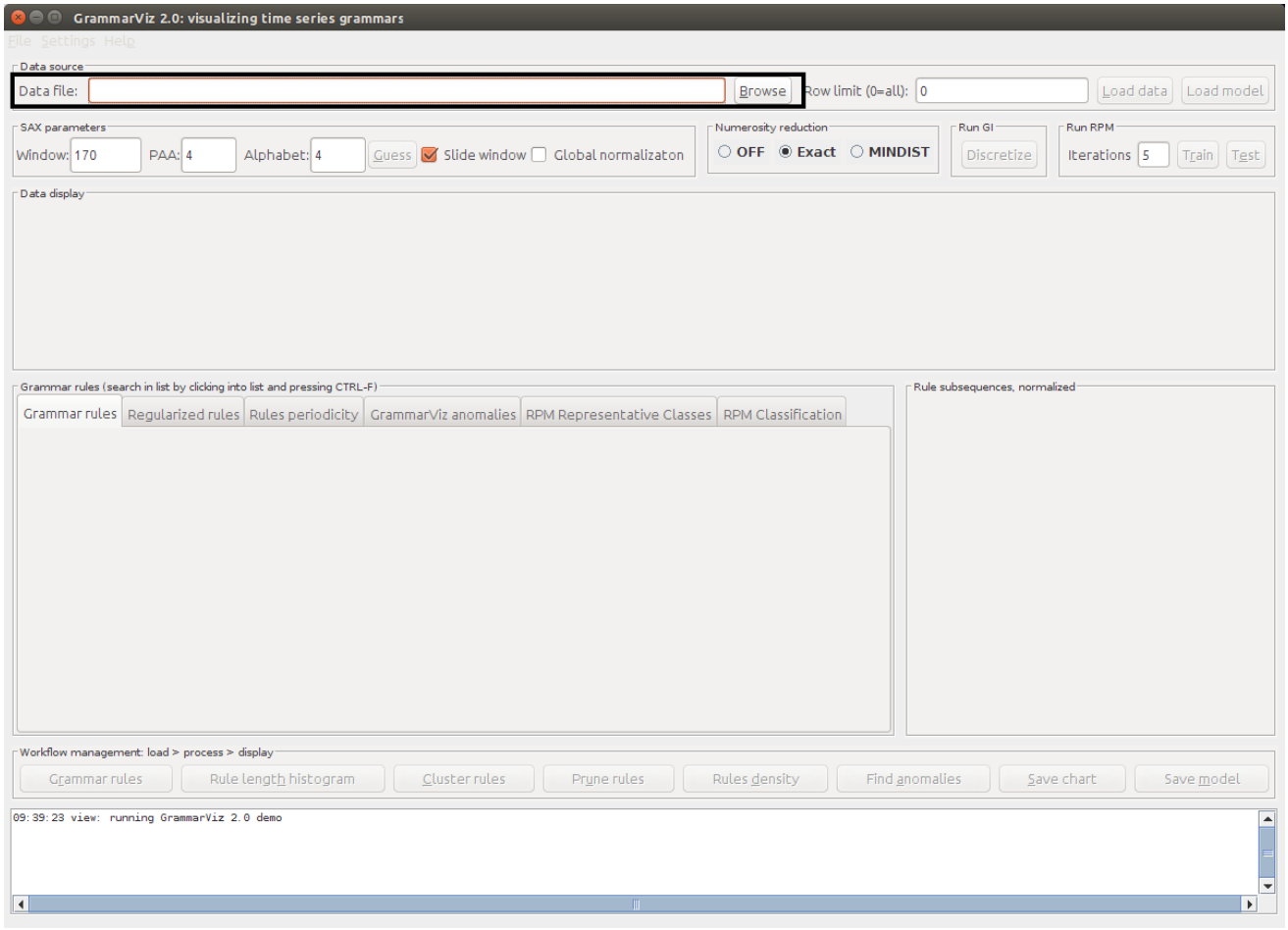

Figure 12: Loading a model

Step 2 This should being up the file browser prompt in figure 13. Using this prompt select the previously saved model.

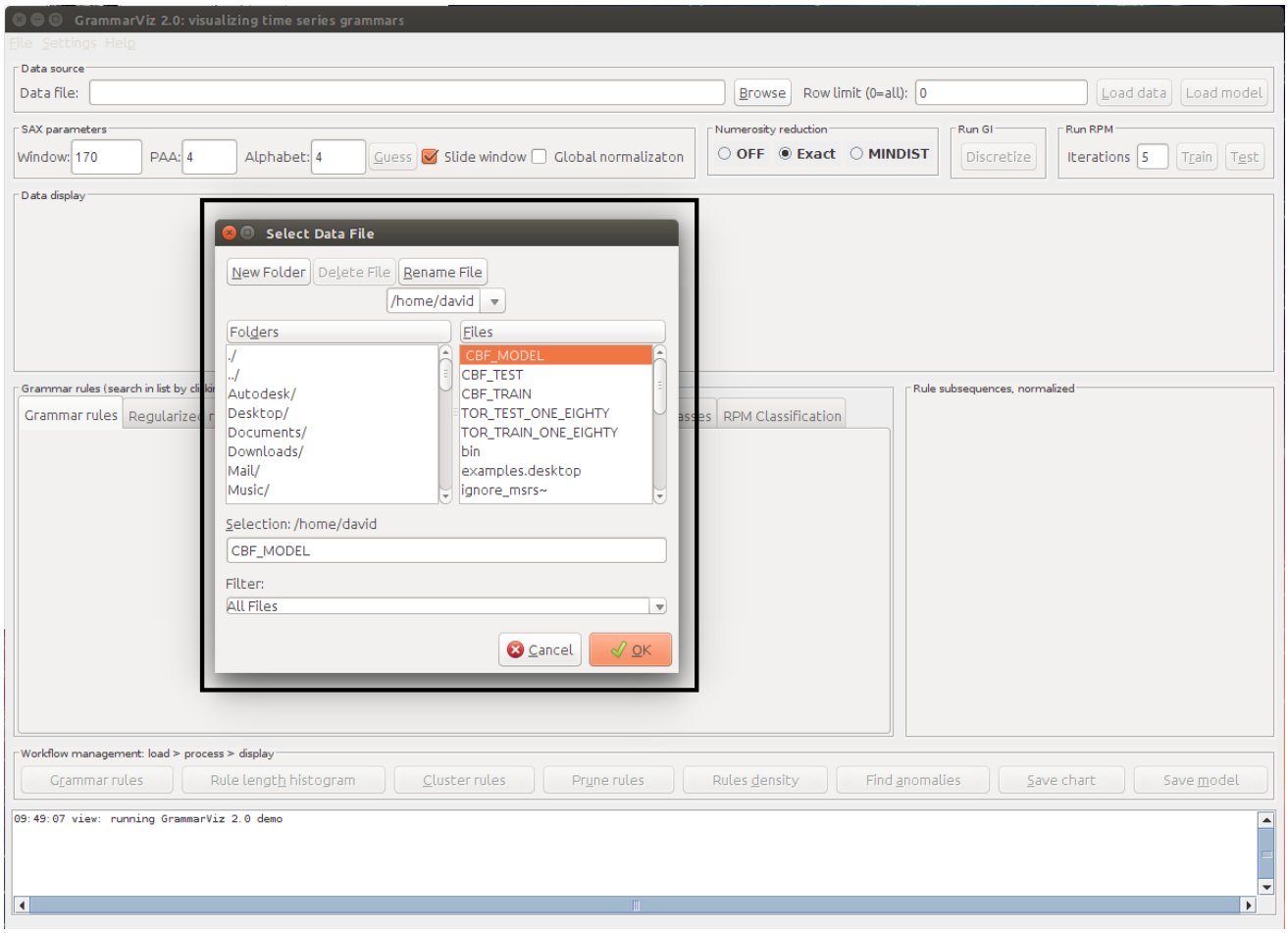

Figure 13: Open the file browser prompt

Step 3 Once the model has been selected, click the "Load Model" button and the model will be loaded into GrammarViz. If the data is not found during the loading step GrammarViz will ask for the location of the data using a file browser prompt, like in figure 15, simple provide the data and the model will finish loading.

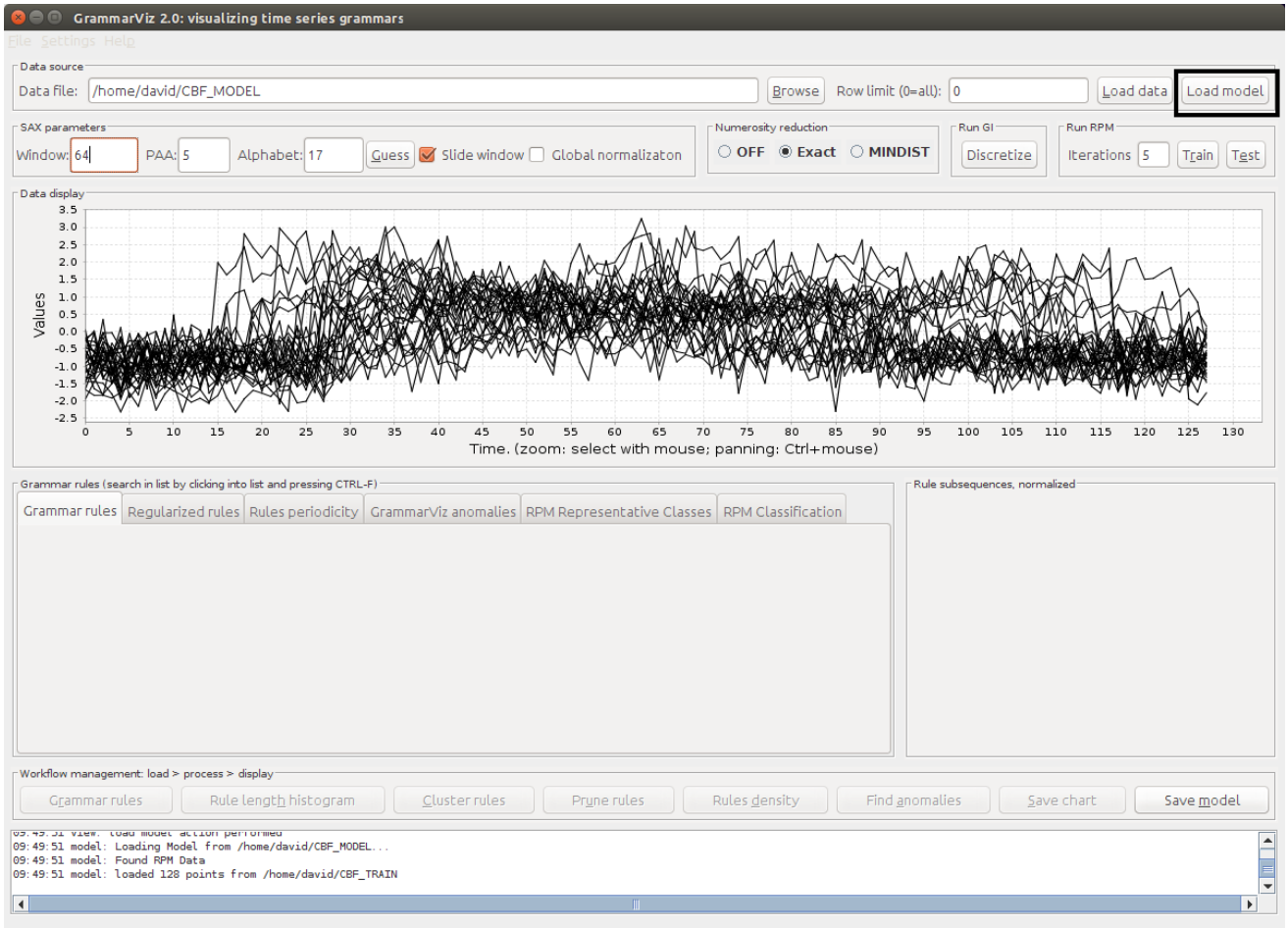

Figure 14: Model loaded

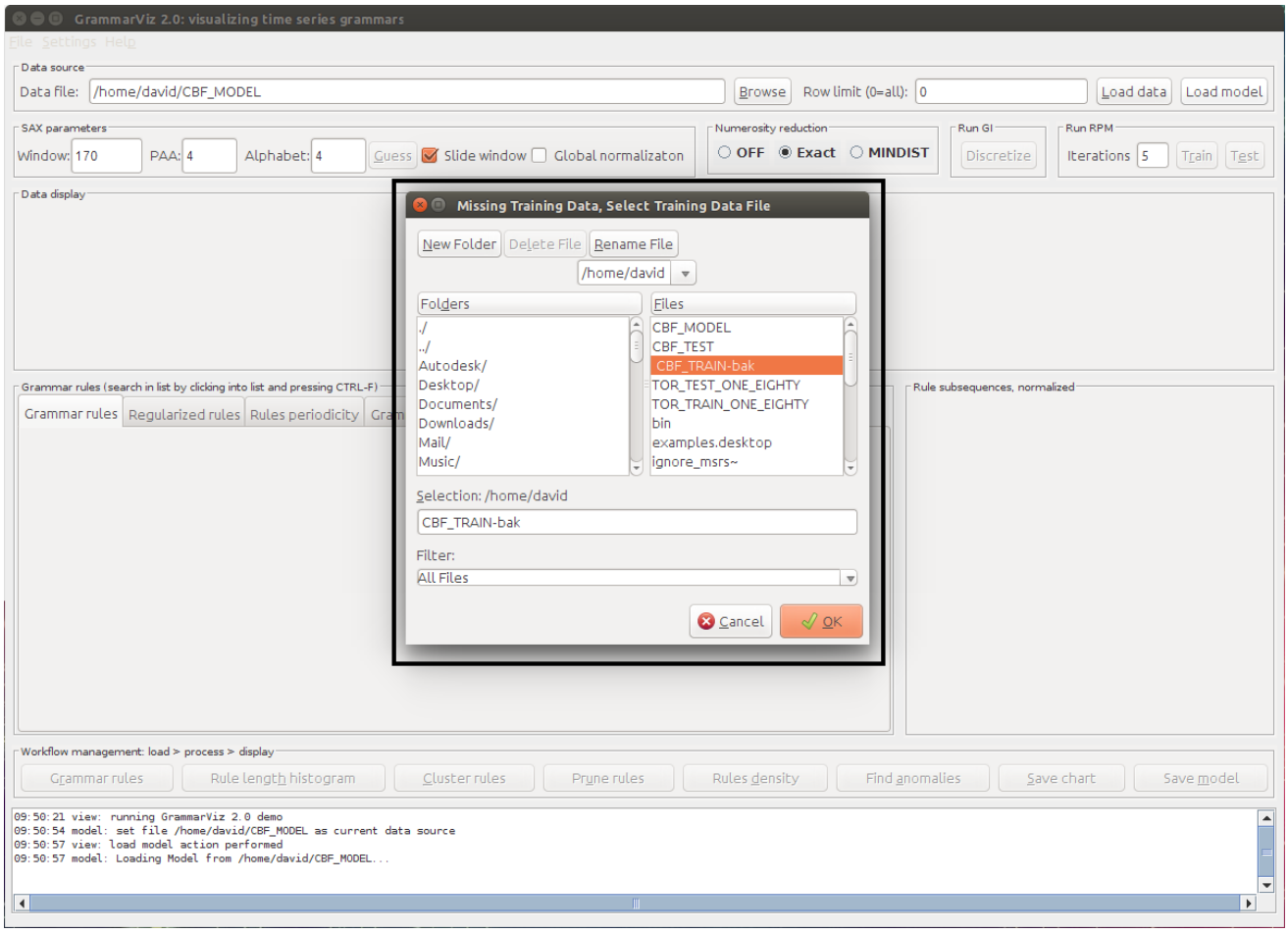

Figure 15: Missing training data file browser prompt

3 RPM settings and what they do

There are a few options that can be changed when using RPM in GrammarViz, some of them have already been mentioned and will be covered again.

3.1 Dynamic Time Warping

Dynamic Time Warping, or DTW, is a method of measuring distance between two time series, this means how similar or different they are to each other. By default RPM uses Euclidean distance which is a simple and fast measurement, however it does not do well when the similar patterns between time series occur at different positions. This is where DTW comes in, it can handle temporal shifts in patterns and, depending on the data, can vastly improve the accuracy of the model. There is a cost however, DTW is a much slower operation and is very expensive to run so it is left as an option for the user.

DTW also has another parameter called "Window" which can have dramatic effects on DTW both in how long it takes to run and its accuracy. The window size basically limits how far DTW will go to try to accurately try to match the two time series. A smaller window will stop DTW from trying to over match them and will take less time to compute. A larger window will take much longer to compute but can allow DTW to match patterns that are father apart. Choosing a good window size can be highly dependent on the data and what is being compared, and therefore some experimentation may be needed to find a good window size. There are a few good rules when choosing a window size, for one a window size greater then 10 will usually give bad results so 10 is considered a good starting point. Often for the more common types of data a 3-5 window size can be much better option with significant speed ups. Note DTW's window should not be confused with the Window size in the SAX parameters section of the main window, these are two different and distinct uses of the word window.

Step 1 To change between Euclidean distance and DTW first open the settings menu:

"Settings"→"GrammarViz options" or press Ctrl+p. This will bring up the settings menu in figure 16.

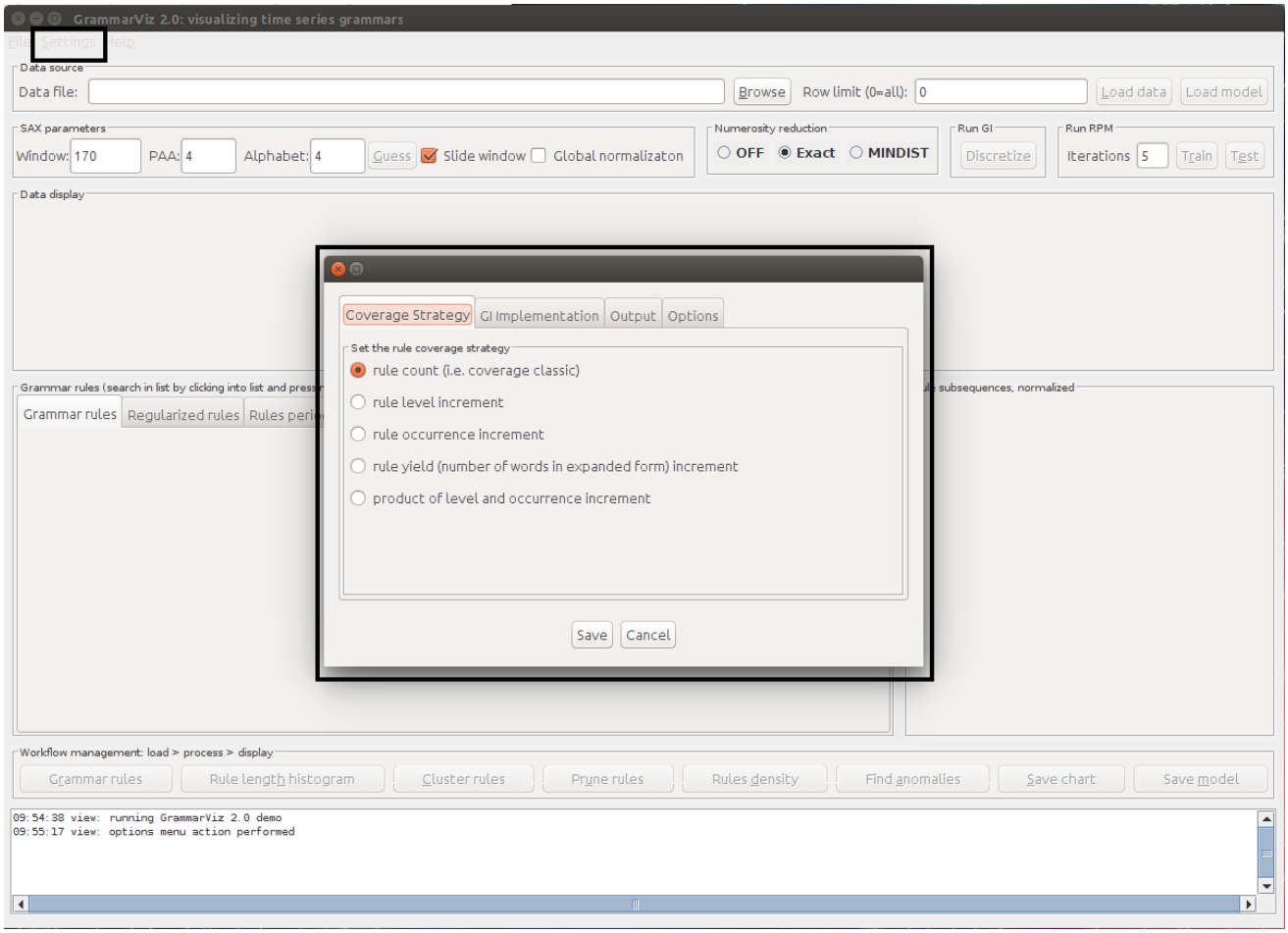

Figure 16: GrammarViz Settings Dialog

Step 2 Now click on the "Options" tab.

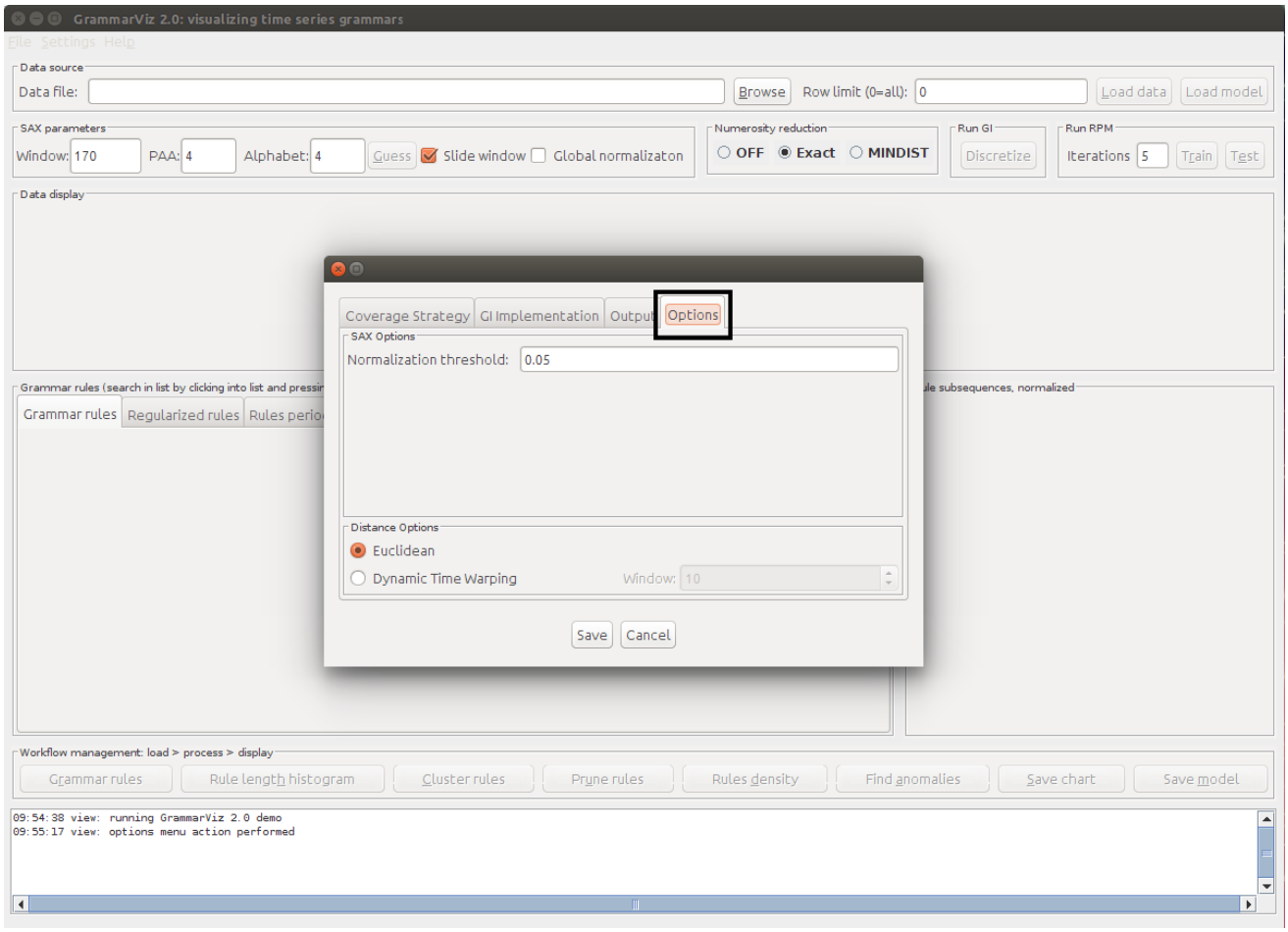

Figure 17: GrammarViz Settings Dialog Options

Step 3 Now select the "Dynamic Time Warping" option and the desired "Window" then click save.

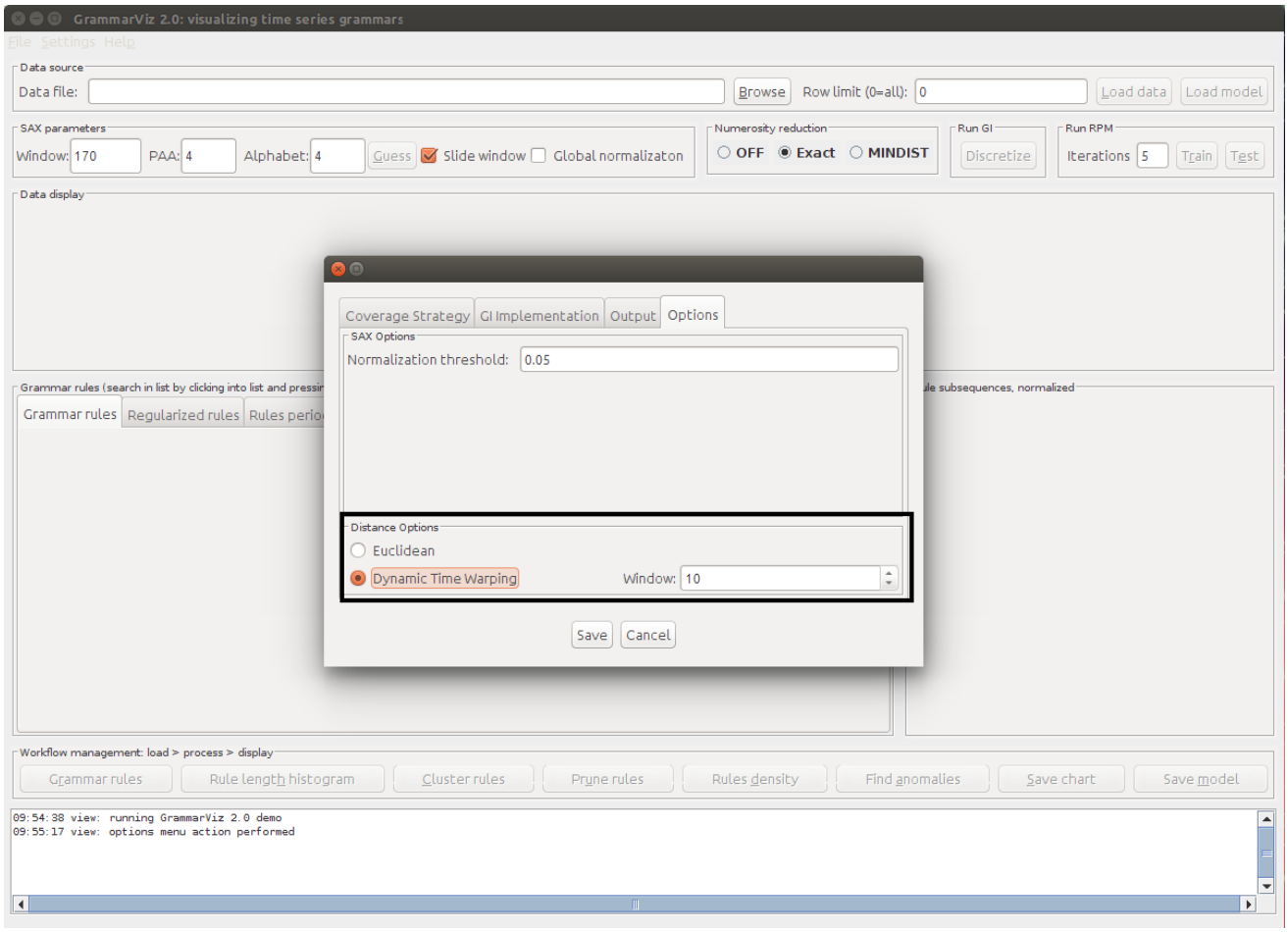

Figure 18: GrammarViz Settings Dialog Options DTW

3.2 Iterations

During the operation of RPM it goes though a step that gets repeated many times. This step only stops under two conditions, a minimum threshold is met or if the maximum number of iterations are reached. The iterations setting found under the "Run RPM" section of the main window in GrammarViz is how the user can control the maximum number of iterations, figure **??**. The number of iterations can have an effect on how accurate the model can get, however the more iterations RPM runs through the longer it will take to complete. This becomes a balance between the quality of the model and the how long the training phase will take. It should also should be noted that RPM can stop before the maximum number of iterations is met if the model has reached an ideal state. However, this does not mean that all models will or even can reach an ideal state before the maximum number of iterations is reached, indeed some data sets may never return a model that meets the requirements. As RPM runs through the iterations the model should get better but the amount it gets better by can be come increasingly insignificant and therefore adding another 10 iterations may not add any significant results to the model. The only way to know if adding more iterations will improve the model is by experimentation which would involve training multiple times, increasing the maximum number of iterations every run until the testing results return no significant improvements.

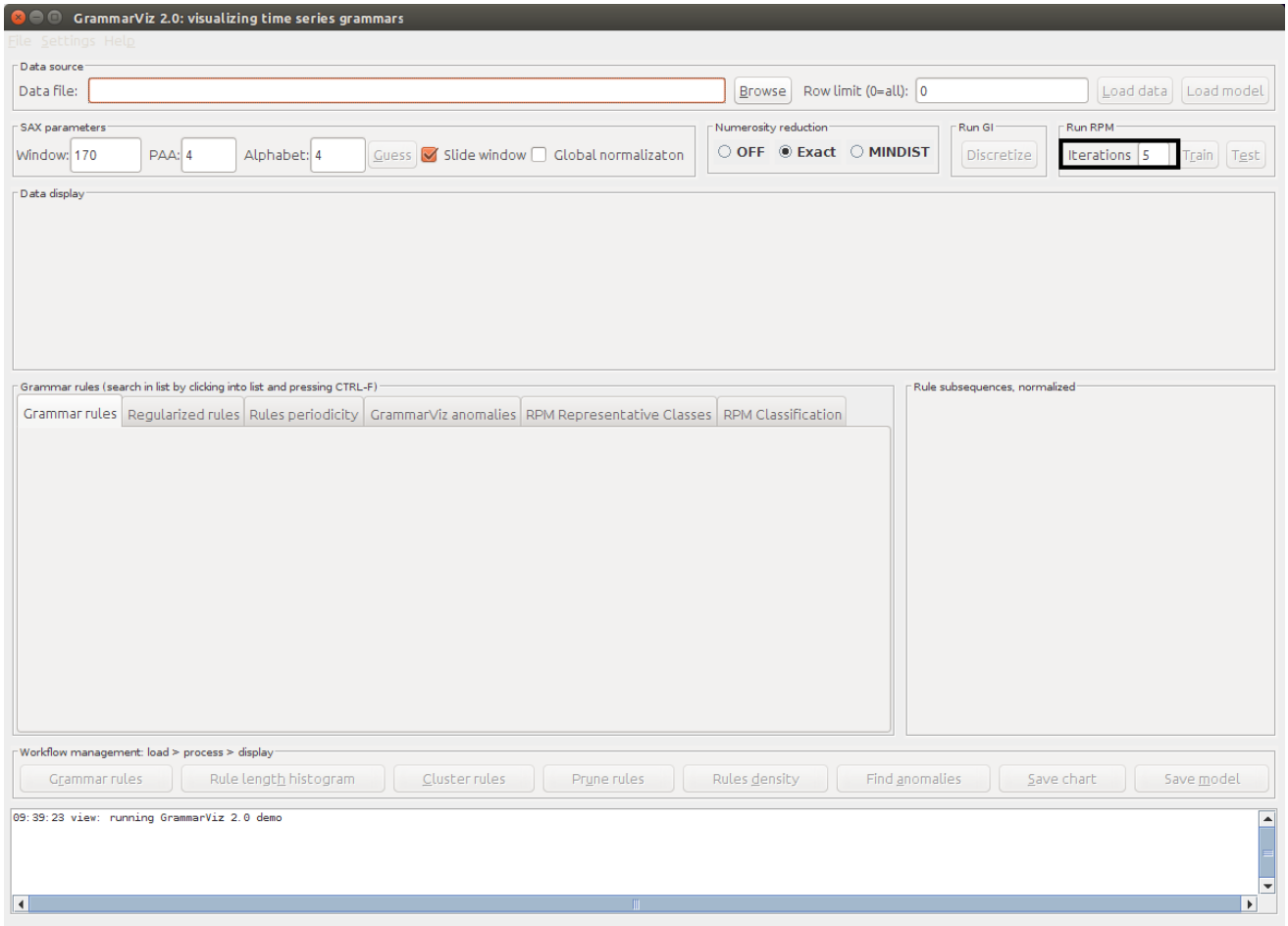

Figure 19: GrammarViz RPM Iteration Setting