Guideline for Training Grobid model for Reference Parsing & Reference Extraction.

Steps for training Reference Parsing model:

This documentation elaborates on the Reference Parsing model.

Refer to following link for the detailed documentation of Grobid model.

[1] <https://grobid.readthedocs.io/en/latest/>

\*\*\*\* The training of both reference parsing and extraction models can only be trained on Linux environment only. The libraries for windows environment are not compatible and a user will not be able to retrain the models.

1. Grobid reference parsing model is a CRF based model which expects the bibliographical elements to be represented in the Text Encoding Initiative ([TEI](http://www.tei-c.org/)) format
2. For preparing the references for the Reference Parsing task following steps should be considered.
   1. The complete reference must be enclosed in **<bibl>&</bibl>** tag.
   2. As per the documentation author information such as “surname” & “give-names” are not considered. Instead **<author>& </author>**  tag is used to represent the sequence of authors found in the reference.
   3. Similarly tags such as **<year>, (<fpage>-<lpage>), <volume>, <issue>** are replaced by **<date>, <biblScope unit=”page”>, <biblScope unit=”volume”>,<biblscope unit=”issue”>**  respectively.
   4. Other bibliographic information which does not belong to the tags mentioned in the page <https://grobid.readthedocs.io/en/latest/training/Bibliographical-references/> must be encoded by **<note> & </note>** tag.
   5. The list of references must be enclosed under **<listBibl> & </listBibl>** tag.
   6. The completed dataset can be splitted into training and testing dataset and needs to be placed appropriately inside the following path assuming the home directory is the folder where the Grobid folder is extracted.  
        
      **/grobid-trainer/resources/dataset/corpus/ - training data.  
      /grobid-trainer/resources/dataset/evaluation/ - testing data**
   7. Execute the following command to initiate the training process  
      **> java -Xmx1024m -jar grobid-trainer/build/libs/grobid-trainer-<current version>-onejar.jar 0 citation -gH grobid-home**
   8. Depending on the number of threads available the training process time can be minimized considerably. Higher the number of threads less is the training time for the model. This can be specified in the file **grobid.properties** found in location /**grobid-home/config/grobid.properties**.  
      Edit the values for property **gorbid.nb\_threads**
   9. Additionally, you also alter the parameters such as **epsilon, window, nbMaxIteration** for the CRF model.
   10. The trained models are created inside the **grobid-home/models/citation/** directory.
3. For using the trained model created in step 2, it is much simple and easy to use GROBID in **service mode**. The API **processCitation** takes exactly a single reference as an input and returns the parsed reference. The time to taken to execute the model on the testing data is very less as the model is loaded only once and served for every request that comes until the local server is restarted.

Additional script can be written to invoke the API and perform it repetitively in loop over the references.

Steps for training Reference Extraction model:

1. Grobid uses a cascading model which process the entire PDF based on the tasks such as identifying sections (header, abstract), parsing references etc. For Reference Extraction, the model **reference-segmenter** is used.
2. For retraining the reference extraction model, the training data that is understood by GROBID needs to be generated by using the **createTraining** batch command. By executing this command on the set of PDF documents, the training data for different models is generated in different format.  
   Each file extension consist of the model name and hence can be easily distinguishable.  
   Moreover, along with the **.tei.xml** format raw data files needs to be retained for training. For e.g. when retraining the **reference-segmenter** model, the data generated by **createTraining**  API will generate files with extension   
   **filename. training.references.referenceSegmenter** - the raw file  
   **filename.training.references.referenceSegmenter.tei** – xml file which can be edited.
3. Is possible that the training data generated will contain mistakes. Hence the generated files (.tei extension) must be evaluated for correctness to prepare a correct training data.
4. If the PDF document contains the references in the footnotes and they have not been identified in the training data generated by **createTraining** API call then, additional steps must be taken to obtain the correct training data.
   1. As mentioned earlier GROBID uses a cascading model sequences, the **segmentation**  model is responsible for identifying the key sections of the document such as Abstract, Header, References.
   2. If the footnote section of the PDF document contains references which were not found in the generated training data, then we need to correct/ introduce **<listBibl> </listBibl>** tag where ever it is required in files that has following extensions.  
      **.segmentation.tei.xml**
   3. We need to retrain the segmentation model first and then execute the **createTraining** API to generate the data.
   4. Inspect the **\*.training.references.referenceSegmenter.tei** if it contains the appropriate data.
   5. Once the process of correction and verification of the training data is complete then it can be moved to the **/grobid-trainer/resources/reference-segmentation/corpus/** folder
   6. Train the model by executing the following command.  
      **> java -Xmx1024m -jar grobid-trainer/build/libs/grobid-trainer-<current version>-onejar.jar 0 reference-segmentation -gH grobid-home**