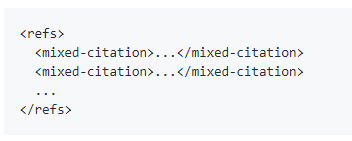
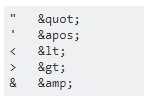
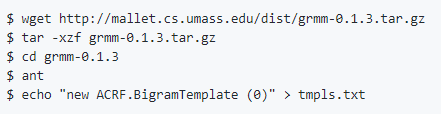
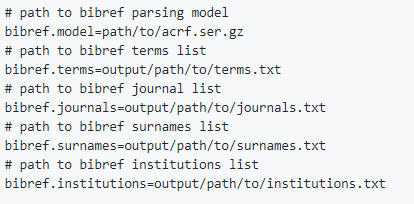
Guideline for Training CERMINE model (Reference extraction & Reference Parsing) on Exgoldstandard.

Most of the points are referred from the Cermine documentation.

Steps for training Reference Parsing model:

1. Obtain the latest version of Cermine code from the Cermine repository.  
   git clone <https://github.com/CeON/CERMINE.git>
2. Obtain the Exgoldstandard dataset.  
   git clone <https://github.com/exciteproject/EXgoldstandard.git>
3. In CERMINE reference parsing is done in three steps: 1) tokenizing the reference string, 2) assigning labels to the individual tokens, and 3) concatenating tokens to form final metadata record. Tokenizing and concatenating steps are straightforward, but assigning labels is a non-trivial problem. In CERMINE it is solved by a supervised classifier, which analyses the tokens represented by bags of features. The features include: terms (tokens themselves), general term classes and orthographic properties [[[1]](#footnote-1)].
4. If the references in the dataset are spanned across multiple files, then they need to be collated into a single training file. Each reference must be placed on **new line** and **enclosed** in **“<refs>” & “</refs>”** tags.  
   
5. Transform the Exgoldstandard dataset into format which is understood by CERMINE model.  
   For this process following guideline needs to be followed.
   1. Each reference string must be written on new line and enclosed in “**<mixed-citation>” & “</mixed-citation>”** tag.
   2. For each sequence of information related to bibliographic element **“<author> & “</author>”** needs to be **replaced** with “**<string-name>” & “</string-name>”.**
   3. For XML special characters found in the reference, they need to be escaped using the standard sequence of characters.  
      
   4. Tag such as **“<other>”** need to be replaced completely.
6. Once the dataset is prepared as per the guideline then we can extract features as represented in the GRMM-specific training format [[2]](#footnote-2) using the following command.  
     
   $ java -cp cermine.jar pl.edu.icm.cermine.bibref. MalletTrainingFileGenerator input/path/to/dataset output/path/to/GRMM/input output/path/to/terms.txt output/path/to/journals.txt output/path/to/surnames.txt output/path/to/institutions.txt  
     
   This will result in two files: a file with features for GRMM (output/path/to/GRMM/input) and additional files with lists of: terms (output/path/to/terms.txt), journals (output/path/to/journals.txt), surnames (output/path/to/surnames.txt) and institutions (output/path/to/insts.txt).
7. Download and build the GRMM library by executing following commands.  
   
8. Now we can initiate the process of training the model for Reference Parsing (refer to following command executed in linux environment).  
   set GRMM variable to the folder containing the GRMM library.  
   “$ GRMM="/export/home/test/Evaluation/Cermine/grmm-0.1.”  
     
   “$ java -cp $GRMM/lib/jgrapht-0.6.0.jar:$GRMM/lib/bsh-2.0b4.jar:$GRMM/class:$GRMM/lib/mallet-deps.jar:$GRMM/lib/grmm.deps.jar edu.umass.cs.mallet.grmm.learning.GenericAcrfTui --training /export/home/test/Cermine/TrainingData/input.txt --testing /export/home/ /test/TestingData/testinput.txt --model-file tmpls.txt > stdout.txt 2> stderr.txt”  
     
   Depending on the size of the training process and the number of interations configured the training phase may last more than 2-3 hours.
9. After the training process is complete “acrf.ser.gz” model is created by the GRMM model. This model along with the term files generated step 6) can be used for testing the model. Before evaluating the learned model we need to make changes in the *application-default.properties* file found under following path   
   “/CERMINE/cermine-impl/src/main/resources/pl/edu/icm/cermine/” .  
     
   Provide appropriate path to the files mentioned in the above snippet.
10. For obtaining the results on the testing dataset following code snippet can be used.  
      
    “BibEntry reference=null;  
    CRFBibReferenceParser parser = CRFBibReferenceParser.getInstance();  
    BibEntryToNLMConverter converter=null;  
     Element element=null;  
     int i=0;  
    String rline=null;  
    ArrayList<String> referenceList=new ArrayList<String>();  
    while((rline=br.readLine()) !=null)  
    {  
    reference = parser.parseBibReference(rline);  
    element = converter.convert(reference);  
    referenceList.add(outputter.outputString(element)); ## referenceList contains the list of parsed references.  
    i=i+1;  
    }”
11. For large testing dataset the above-mentioned code snippet can be used in loop for each file containing the references.

Steps for training Reference Extraction model:

1. For reference extraction process Cermine uses SVM classifier to identify the different zones present in the PDF document. For this purpose, it is necessary to correctly label sections of document that needs to be considered
2. As the first step in the training of reference extraction process, it is required to generate the appropriate TrueViz file for each PDF document. The TrueViz file contains the structure information which were generated automatically by Cermine and has a possibility of containing error i.e. zones being misclassified or not classified at all. Cermine, has provided an additional tool SegmEdit tool [[3]](#footnote-3) which can be used to correct the TrueViz files and thus create and correct training data for the classifier.
3. Please refer to the installation guidelines for SegmEdit tool which requires wxgtk 3.8 wx widget for its proper functioning. This widget can be downloaded from following link  
   [https://www.wxwidgets.org/downloads/].  
   Verify the installation of this package and see for the version of the package. The following code snippet will help in identifying the appropriate version.   
   For the purpose of evaluation we had used version 2.8.xx .  
   “import wx  
   wx.version()”
4. Once installed just run the ./segmedit.py file and provide the path for the PDF files and its corresponding TrueViz files.
5. Verify the labels for each section of the PDF document rendered by the Segmedit tool.
6. For references in footnotes please select the section in the document which contain the footnotes and label it with appropriate label.
7. Repeat the process of verifying the document for all the training data.
8. Once complete execute the following commands.

$ java -cp cermine.jar pl.edu.icm.cermine.libsvm.training.SVMMetadataBuilder -input path/to/directory/with/trueviz/ -output model-metadata

$ java -cp cermine.jar pl.edu.icm.cermine.libsvm.training.SVMBodyBuilder -input path/to/directory/with/trueviz/ -output model-body

$ java -cp cermine.jar pl.edu.icm.cermine.libsvm.training.SVMInitialBuilder -input path/to/directory/with/trueviz/ -output model-category.

1. During the training process, CERMINE reads TrueViz files from the input directory, converts all zones to feature vectors and generates an SVM model from the entire set. The output contains two files per classifier: a model (in our case: model-metadata, model-body and model-category files) and a file containing values ranges of the features (in our case: model-metadata.range, model-body.range and model-initial.range files). Both files are used by the classifiers during analysing the document.
2. Provide the path to the model files generated in the ***application.properties*** file.

1. https://github.com/CeON/CERMINE/blob/master/TRAINING.md [↑](#footnote-ref-1)
2. https://github.com/CeON/CERMINE/blob/master/TRAINING.md [↑](#footnote-ref-2)
3. https://github.com/CeON/SegmEdit [↑](#footnote-ref-3)