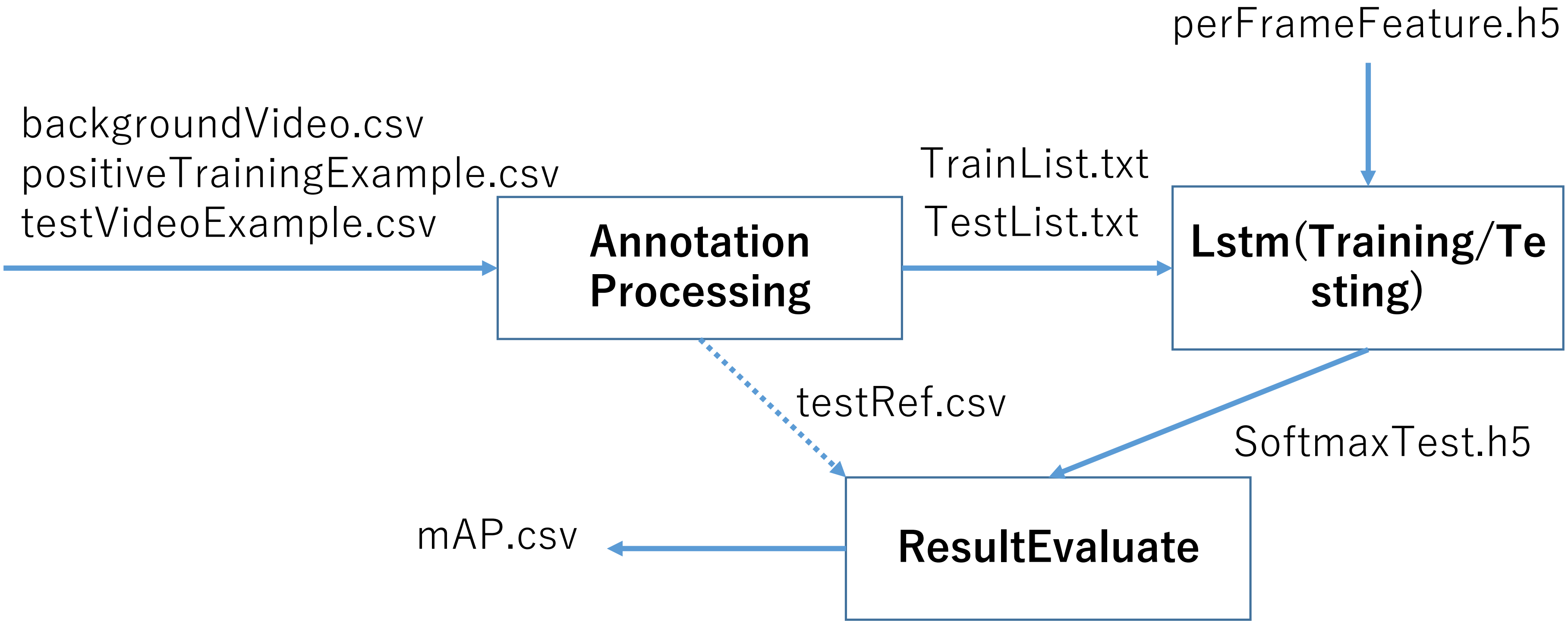
How to run Lstm Parts in MED baseline

By Mengxi,
Shinoda Lab.

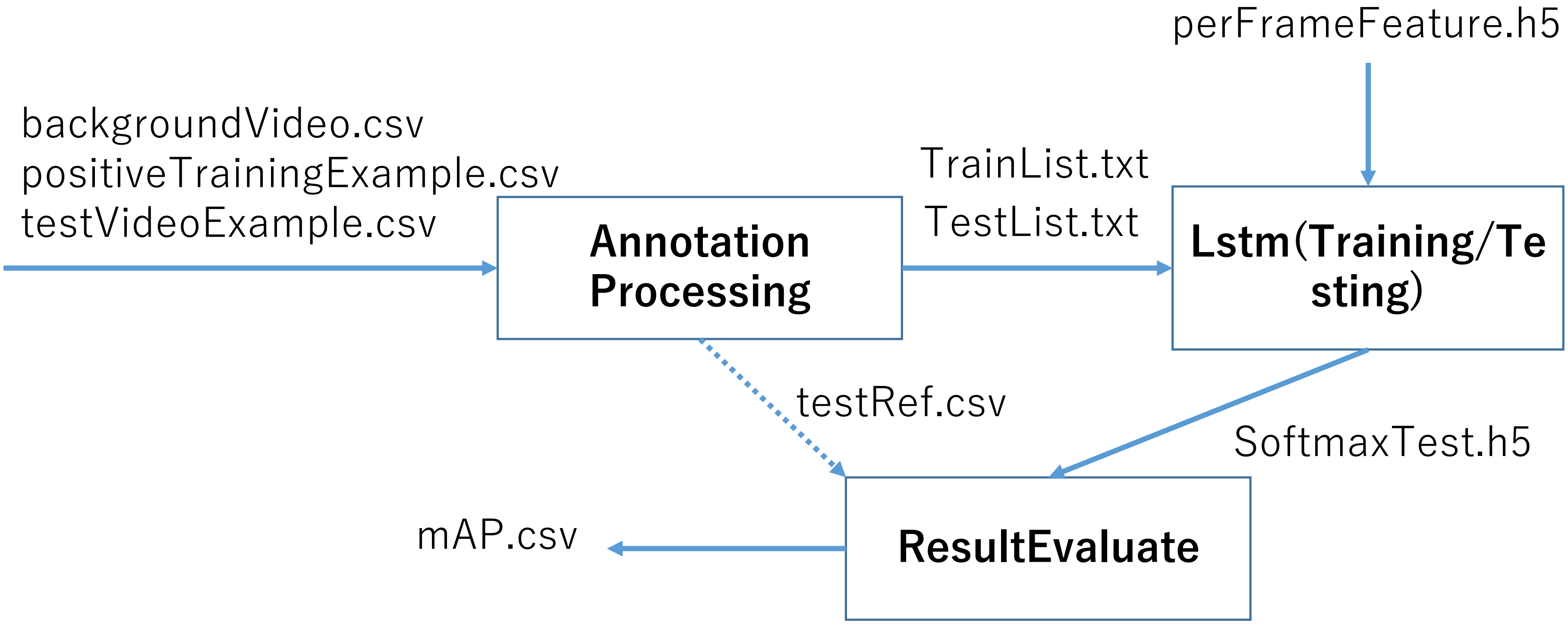
LSTM Parts Overview



Three Components:

- **AnnotationProcessing** (C++): convert the training and testing annotations (*.csv) to the annotationFiles ({Train,Test}List.txt) that can be fed to the **Lstm** Module. Optionally, you can create a new testRef.csv file corresponding to your TestList.txt for later use in **ResultEvaluate** module.

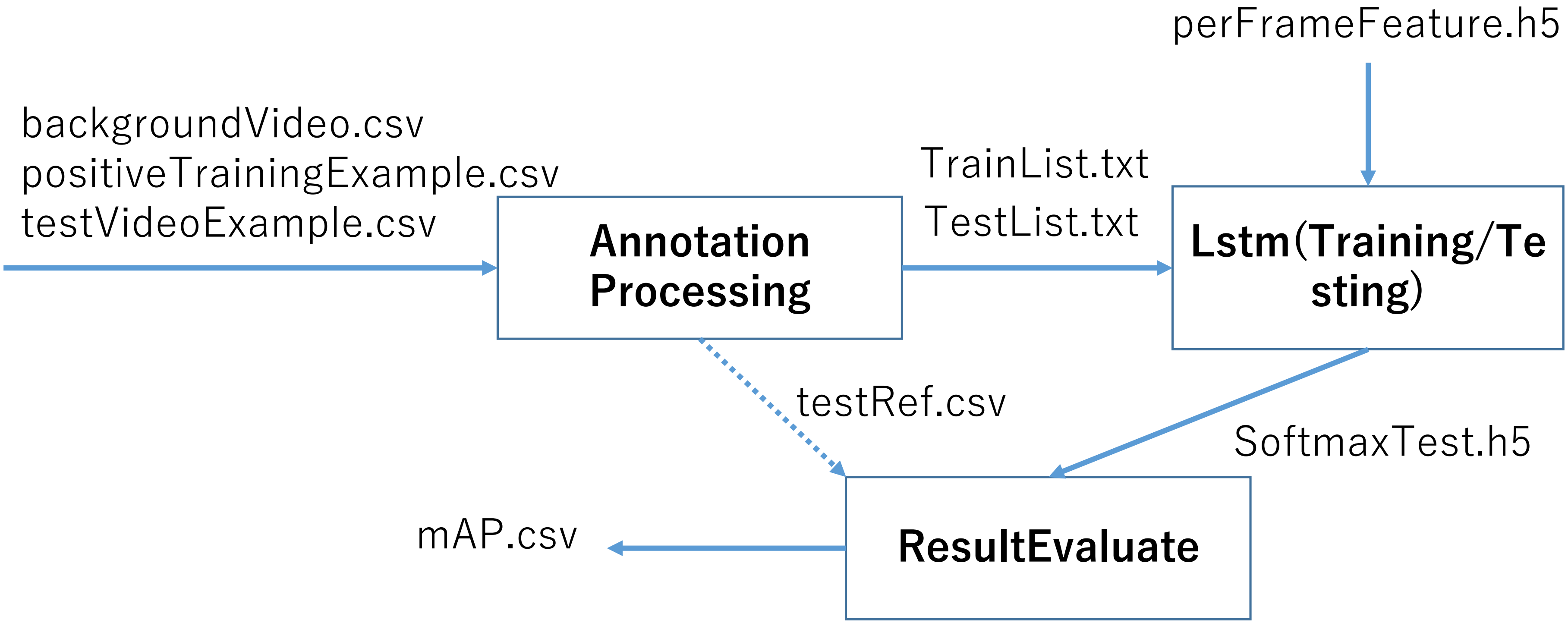
LSTM Parts Overview



Three Components:

- **Lstm** (Torch/Lua): Train LSTM models and test on testing data. Eventually we can get a bunch of LSTM models and Softmax probability output on the testing data. The Softmax probs can be fed into **ResultEvaluate** module to evaluate the AP (Average Precision) performance of the trained LSTM models.

LSTM Parts Overview



Three Components:

- **ResultEvalute** (bash, Python): convert the Softmax probs to the ranking list and evaluate the AP (Average Precision).

AnnotationProcessing

1. Enter the AnnotationProcessing directory, you can find:
   - (source codes) BasicOperations/
   - (source codes) src/
   - (compiling bash) compile.sh
   - (starter bash script) convertCsvToTxt.sh

2. Execute the ‘compile.sh’ to compile the C++ source codes
   - ./compile.sh
   - Note: The source code depends on the ‘boost’ library. If you are in the Tsubame3, the ‘boost’ lib has already be installed.
   - After compilation, you can find an executable named ‘convertCsvToTxt’

AnnotationProcessing

3. Open the bash script ‘convertCsvToTxt.sh’ and edit it.
    - Input Params:
        - TRAIN_BG_FILE_PATH: background csv for training
        - TRAIN_EVENT_FILE_PATH: the positive examples csv for training
        - FEATURE_DIR_NUM: number of feature directories
        - FEATURE_DIR_PATH: the feature directory paths
        - EVENT_ID_OFFSET: the offset between event Id and 0
        - EVENT_NUM: the number of events
        - IS_OMMIT_BACKGROUND: set to 0 if you want to include background videos for training
    - Output Params:
        - TRAIN_TXT_PATH: the ‘txt’ format annotations for LSTM training
        - TEST_TXT_PATH: the ‘txt’ format annotations for LSTM testing
        - NEW_TEST_REF_FILE: the test ‘csv’ file that can be used for final evaluation in ‘ResultEvaluate’ part

AnnotationProcessing

3. Open the bash script ‘convertCsvToTxt.sh’ and edit it.
   - **If you are in Tsubame3, you don’t need to edit all the Input params. You only need to edit the output params to the paths you have the write permission on.**

4. Run the ‘convertCsvToTxt’
   - Run by issuing ./convertCsvToTxt.sh
   - After running, you are expected to find ‘Train.txt’, ‘Test.txt’ and ‘testRef.csv’ in the places you have specified in Step 3.

Lstm

1. Enter the Lstm directory, you can find:
    - (lua files) Many Lua files that are for training and testing LSTMs with torch
    - (starter bash scripts) trainStarter.sh, testStarter.sh and testStarterBatch.sh
2. For training, open the trainStarter.sh and edit it.
    - Input params:
        - TRAIN_ANNOTATION_PATH: the training ‘txt’ annotation file you created in ‘AnnotationProcess’ part
        - DATA_SET_NAME: Dataset name of feature hdf5
        - Setting of the network: INPUT_DIM, OUTPUT_DIM and HIDDEN_DIM
        - Setting of the optimization: MODEL_SAVING_STEP, EPOCH_NUM, BATCH_SIZE, LEARNING_RATE, LEARNING_RATE_DECAY, WEIGHT_DECAY, GRADIENT_CLIP
        - Setting of GPU: GPU_ID
    - Output param:
        - MODEL_SAVING_DIR: the directory you will save your models

Lstm

2. For training, open the trainStarter.sh and edit it.
    - If you are in the Tsubame3 and want to train the LSTM using default parameters, you only need to edit the following params:
        - TRAIN_ANNOTATION_PATH: specify the path of training ‘txt’ annotation file you created in ‘AnnotationProcess’ part
        - MODEL_SAVING_DIR: specify the places you’d like to save your models

3. You may want to submit the training as a job to Tsubame3. The bash script ‘submitLstmTrain.sh’ can help.
    - Open the ‘submitLstmTrain.sh’ and change the output and error output options (etc. –o and –e options)
    - Submit the job by issuing qsub –g YOUR_GROUP submitLstmTrain.sh

4. After the job finishes (around 1 hour), you can find the Lstm models in the place you specify in Step 2.

Lstm

5. For testing, first edit the testStarter.sh
   - Input params:
     - TEST_ANNOTATION_PATH: the testing ‘txt’ annotation file you created in ‘AnnotationProcess’ part
     - DATA_SET_NAME: Dataset name of feature hdf5
     - MODEL_PATH: the Lstm model you want to test. Here we set it to $1 for calling from ‘testStarterBatch.sh’
     - BATCH_SIZE
     - Setting of GPU: GPU_ID. Here we set it to $3 for calling from ‘testStarterBatch.sh’
   - Output param:
     - OUTPUT_PATH: the place to output the video level softmax probability. Here we set it to $2 for calling from ‘testStarterBatch.sh’
   - If you are in Tsubame3, you only need to edit:
     - TEST_ANNOTATION_PATH: specify the path of testing ‘txt’ annotation file you created in ‘AnnotationProcess’ part

Lstm

6. For testing, then you edit testStarterBatch.sh. This is a wrapper of ‘testStarter.sh’. It can help you to test all the models in a directory, instead of only one model as in ‘testStarter.sh’

   - Input params:
     - MODEL_DIR: the directory your Lstm models are located in.
     - Setting of GPU: GPU_ID
   - Output param:
     - OUTPUT_DIR: the place you want to put the softmax probability for each model being tested.

   - If you are in Tsubame3, please edit:
     - MODEL_DIR: this can be the one you specify in the training
     - OUTPUT_DIR: specify the place you want to store the softmax probability for the models

Lstm

7. You may want to submit the testing as a job to Tsubame3. The bash script ‘submitLstmTest.sh’ can help.
    - Open the ‘submitLstmTest.sh’ and change the output and error output options (etc. –o and –e options)
    - Submit the job by issuing qsub –g YOUR_GROUP submitLstmTest.sh

8. After the job finishes (around 20 mins), you can find the softmax probability in hdf5 format in the place you specify in Step 6.

ResultEvaluate

1. For evaluating the detection results in AP (average precision), enter the directory ResultEvaluate. You can find:
   - (bash script) evaluateStarter.sh
   - (bash script) evaluateLstmDetectionResults.sh
   - (python script) convertH5SoftmaxToCsv.py
   - (bash script) ap.sh
2. You only need to edit evaluateStarter.sh
   - InputParam:
     - H5_SOFTMAX_DIR: the softmax directory containing the video level softmax probs output by ‘Lstm/test’
     - TEST_EVENTDB: the event specification in ‘csv’
     - TEST_REF: the annotation ‘testRef.csv’ output by the ‘AnnotationProcessing’ part
     - EVENT_ID_OFFSET: the offset between event Id and 0
     - IS_H5_INCLUDE_BACKGROUND: set to 1 if the training includes the background examples
   - OutputParam:
     - OUTPUT_AP_DIR: the directory to store AP.

ResultEvaluate

3. If you are in Tsubame3, you only need to edit the following parameters:
   - H5_SOFTMAX_DIR: this can be the directory you specify for storing softmax probs when you test Lstm.
   - TEST_REF: this can be the ‘testRef.csv’ file output by ‘AnnotationProcessing’ part.
   - OUTPUT_AP_DIR: you should specify the place to store the ap result.
4. Run evaluateStarter.sh.
   - Issue ./evaluateStarter.sh
   - You can check the ap in ${OUTPUT_AP_DIR}. You are expected to get mAP about 0.43 around epoch 33.

That’s all for Lstm Parts.