Master IARFID
Reconocimiento de Escritura (RES)
Handwritten Text Recognition (HTR)
Practical session: State-of-the-art HTR systems
Training

PRHLT-Group

Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València

Description

- Deep Neural Networks are the state-of-the-art technology for handwritten text recognition.
- We will made a complete HTR system based on neural networks.
- First, we will train an optical model based on Deep Neural Networks with Connectionist Temporal Classification (CTC) and then, we will decode the test set without language model.
- Secondly, we will train an n-gram language model to made a complete HTR system based on Weighted Finite-State Transducers (WFST) and then, we will decode and get word-graphs from the test set.

Training the optical model

1. Line image preprocessing and getting features.
   - Get the preprocessing tool

```
wget --no-check-certificate \
http://www.prhlt.upv.es/~mpastorg/RES/linePreprocess.cc
```

   - Compile it:

```
g++ -o linePreprocess linePreprocess.cc -I. \
`pkg-config --cflags --libs opencv` -O3
```

   - Process it:

```
 mkdir data/feat

for file in `ls data/Corpus_clean_lines/*`; do
   NOM=`basename $file`
   echo $file
  ./linePreprocess -i $file -o data/feat/$NOM
done
```

Training the optical model

2. Preparing transcriptions

```
mkdir data/text

  for f in data/txt/*; do
   awk -v fileName=$(basename $f .txt) '{
      printf("%s_%i ", fileName, NR-1);
      gsub("[␣_]"," ")
      print
   }' $f;
done | sed -e "s/\s+/ /g" \
         -e "s/^\s+//g" \
         -e "s/[\$\t]/ /g" > data/text/transcriptions.txt
```

“<WARNING>”: Search the transcripts for the special character '␣' and change it in the terminal. When copying it is not pasted.

Training the optical model

3. Getting the symbol list

```
mkdir -p data/lists

awk '{
  for(i=2;i<=NF;++i) {
    for(j=1;j<=length($i);++j)
       print substr($i, j, 1);
  }
}' data/text/transcriptions.txt |\
LANG=C sort -u > data/lists/symbols.lst
```

4. Getting the maping between characters and numbers

```
awk '{print $1,tolower($1)}' \
data/lists/symbols.lst > data/lists/unitsMap.lst
```

Training the optical model

5. Simplifying the set of optical units
   - Edit data/lists/unitsMap.lst and change, or delete the symbols on the second column
   - Units in the first column will be used to replace the ones in the second when we normalize the ground thruth.
   - If there is not a symbol on the second column, it will be deleted on the ground truth.
   - Leave just ASCII caracters on the second column or empty.

Training the optical model

6. Getting the ground truth at optical units level

```
sed -e "s/\^{\S*}//g" -e "s/[[][^]]*[]]//g" \
    data/text/transcriptions.txt | \
awk -v MAP=data/lists/unitsMap.lst 'BEGIN{
  while ((getline < MAP) > 0) {
    if (NF == 2) DICT[$1] = $2
   }
   DICT[" "] = " ";
}{
   printf ("%s ",$1)
   for(w=2; w<=NF; w++) {
     N=split(tolower($w),CHAR,"")
     for(i=1; i<=N; i++)
       if (CHAR[i] in DICT)
         printf("%s ",DICT[CHAR[i]])
     printf ("<space> ")
   }
   printf("\n")
}' > data/text/transcriptions_char.txt
```

Training the optical model

7. Getting partitions: training, validation and test files list;
   - training files list

```
for file in `cut -d" " -f1 data/text/transcriptions.txt`; do
      [ -e data/feat/${file}.jpg ] && echo $file;
  done| head -15000 > data/lists/train.lst
```

   - validation files lists

```
for file in `cut -d" " -f1 data/text/transcriptions.txt`; do
      [ -e data/feat/${file}.jpg ] && echo $file;
  done| head -16000 | tail -1000 > data/lists/val.lst
```

   - test files list

```
for file in `cut -d" " -f1 data/text/transcriptions.txt`; do
      [ -e data/feat/${file}.jpg ] && echo $file;
  done| tail -3359 > data/lists/test.lst
```

Training the optical model

8. Getting the ground truth for training and validation at optical model level

```
for part in train val test; do
    awk -v ListFiles=data/lists/${part}.lst '
      BEGIN{
          while ((getline file < ListFiles) > 0) FILES[file]=1
    }{
          if ($1 in FILES) print
    }' data/text/transcriptions_char.txt \
        > data/text/${part}_char.txt
done
```

9. Getting the optical symbols list to train adding some special ones.

```
  sort -u data/lists/unitsMap.lst | awk 'BEGIN{
        cont=0;
        print "<eps>", cont++;
        print "<ctc>", cont++;
        print "<space>", cont++;
    }{if (NF==2 && !($2 in AP)) {AP[$2]=1; print $2,cont++}}' \
  > data/lists/symbols_train.lst
```

Training the optical model

10. Tool to ctc train
    - Set the virtual environtment and libraries required

```
python3.6 -m venv RDNN-HTR-PY
source RDNN-HTR-PY/bin/activate
```

    - Get the PyLaia

```
git clone https://github.com/jpuigcerver/PyLaia.git
cd PyLaia
pip3.6 install -r requirements.txt
python3.6 setup.py install
cd ..
```

Create the optical model

11. Define the model parameters: convolutional layers, kernel size, maxpool size, number of features, recurrent layers, type, number of layers and units, ...

```
mkdir -p models/Optical

pylaia-htr-create-model \
   --print_args True --logging_level info \
   --train_path ./models/Optical \
   --model_filename Rodrigo.net \
   --fixed_input_height 64 \
   --cnn_kernel_size 3 3 3 3 \
   --cnn_dilation 1 1 1 1 \
   --cnn_num_features 12 24 48 48 \
   --cnn_batchnorm True True True True \
   --cnn_activations LeakyReLU LeakyReLU LeakyReLU LeakyReLU \
   --cnn_poolsize 2 2 0 2 --use_masked_conv=true \
   --rnn_type LSTM --rnn_layers 3 \
   --rnn_units 256 --rnn_dropout 0.5 \
   --lin_dropout 0.5 1 data/lists/symbols_train.lst
```

Train the optical model with ctc

11. Define the training parameters: batch size, early stop epochs, learning rate, . . .

```
pylaia-htr-train-ctc \
--print_args True \
--show_progress_bar True \
--logging_level info \
--logging_also_to_stderr info \
--logging_file ./models/Optical/train-crnn.log \
--train_path ./models/Optical \
--model_filename Rodrigo.net \
--batch_size 10 \
--learning_rate 0.0003 \
--use_distortions True \
--max_nondecreasing_epochs 60 \
--delimiters="<space>" \
data/lists/symbols_train.lst data/feat \
data/text/train_char.txt data/text/val_char.txt
```