ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

Documentazione smallNorbCreator v.0.4

Vincenzo Lomonaco

May 16, 2015

1 INTRODUZIONE

Il progetto *smallNorbCreator* ha come obiettivo la creazione di un meta-database di sequenze di immagini a partire dal database *"small NORB v.1.0"* orginale. Ad oggi è composto da 9 sorgenti java ed un jar eseguibile:

1. NorbConverter.java
2. NorbSampler.java
3. NorbSamplerTrain.java
4. NorbSamplerTest.java
5. NorbSeqExplorer.java
6. NorbKNN.java
7. KNNTestSeries.java
8. SeqVerifier.java
9. NorbCreator.java
10. norbCreator.jar

2 NORBCONVERTER.JAVA

NorbConverter.java si occupa della conversione delle immagini originali dal formato matlab a quello bitmap eseguendo eventualmente un sottocampionamento delle immagini e dividendole per classi in sottocartelle differenti.

3 NORBSAMPLER.JAVA

NorbSampler.java è una classe astratta che viene ereditata dalle sottoclassi *NorbSamplerTrain.java* e *NorbSamplerTest.java*. Essa si occupa di implementare le funzionalità di base offerte da un generatore di sequenze di base, a prescindere esse siano di train o di test.

4 NORBSAMPLERTRAIN.JAVA

NorbSamplerTrain.java si occupa del sampling del training set. Innanzitutto si parte dal presupposto che training set e test set siano riuniti in un unica cartella contenente le 5 sottocartelle rispettive a ciascuna classe 0,1,2,3,4. Per ogni oggetto presente in queste directory una o più sequenze di immagini possono essere create. Le sequenze hanno lunghezza arbitraria e sono caratterizzate da una singola variazione unitaria di frame in frame in una sola delle dimensioni possibili tra elevazione, azimuth e illuminazione. Ognuna di queste possibili variazioni ha una certa probabilità. Ad esse si aggiunge quella del cambiamento di direzione delle variazioni. Alla fine tutti i parametri di cui sopra e le sequenze di immagini vengono scritte all'interno di un'unico file di testo.

4.1 IL FORMATO DEL FILE DI OUTPUT

il formato per il file di output del training set che viene creato una volta lanciato il programma è un file di testo semplice la cui intestazione contiene le seguenti informazioni:

- *Config Type*: Tipo del file di configurazione, in questo caso "Train".
- *nClass*: numero di classi da considerare (vengono prese in considerazione le classi da indice 0 a n-1).
- *nObjxClass*: numero di oggetti da considerare per classe (vengono prese in considerazione gli oggetti da indice 0 a n-1).
- *nSeqxObj*: numero di sequenze di immagini da creare per ciascun oggetto.
- *elevationProb*: Probabilità che la variazione tra un frame e l'altro riguardi l'elevazione (un incremento o decremento della dell'elevazione ha uuguale probabilità).
- *azimuthProb*: Probabilità che la variazione tra un frame e l'altro riguardi l'azimuth (un incremento o decremento della dell'azimuth ha uguale probabilità).

- *lightingProb*: Probabilità che la variazione tra un frame e l'altro riguardi la luminosità (un incremento o decremento della dell'illuminazione ha uguale probabilità).
- *flipProb*: Probabilità di cambiare direzione di variazione.
- *seqLen*: Lunghezza di ciascuna sequenza.
- *seed*: Seme per il generatore di numeri pseudo-casuali.

In Fig. 4.1 è possibile osservare un esempio di intestazione del file di configurazione generato a seguito dell'esecuzione dello script *NorbSamplerTrain.java*. In seguito, precedute da una linea vuota e da un'intestazione specifica della sequenza vi sono tutte le immagini costituenti la stessa identificate dal loro nome. L'intestazione contiene gli indici rispettivi alla classe, all' oggetto ed alla sequenza, come è possibile osservare in Fig. 4.1.

```
Config Type: Train
-------------------------
nClass: 2
nObjxClass: 10
nSeqxObj: 1
ElevationProb: 0.3
AzimuthProb: 0.3
LightingProb: 0.2
FlipProb: 0.2
seqLen: 20
seed: 1234
-------------------------
```

Figure 4.1: Un esempio di intestazione del file di configurazione per il training set.

```
---------------
Class:    0
Object:   0
Sequence: 0
---------------
00_02_10_05.bmp
00_03_10_05.bmp
00_03_12_05.bmp
00_04_12_05.bmp
00_05_12_05.bmp
00_06_12_05.bmp
00_07_12_05.bmp
...
```

Figure 4.2: Un esempio di sequenza con relativa intestazione.

5 NORBSAMPLERTEST.JAVA

NorbSamplerTest.java si occupa del sampling del test set. Leggendo i dati presenti nel file di configurazione del training set, esso genera per ogni oggetto una sequenza di immagini della stessa lunghezza di quelle create nel training set ma in cui ogni frame ha una distanza minima paramentrica da quelle utilizzate nel training set per lo stesso oggetto. In particolare la distanza tra due frame è calcolata come la somma delle variazioni su ciascuna dimensione. L'unico paramentro tarabile che differisce dal training set è dunque il numero di sequenze da creare per ciascun oggetto. Anche in questo caso alla fine tutti i parametri, le sequenze di immagini e la distanza media dal training set, vengono scritte all'interno di un'unico file di testo (vd. *test_conf.txt* in allegato).

5.1 IL FORMATO DEL FILE DI OUTPUT

Anche il formato per il file di output del test set che viene creato una volta lanciato il programma *NorbSamplerTest.java* è un file di testo semplice la cui intestazione contiene le seguenti informazioni:

- *Config Type*: Tipo del file di configurazione, in questo caso "Test".
- *nClass*: numero di classi da considerare (vengono prese in considerazione le classi da indice 0 a n-1).
- *nObjxClass*: numero di oggetti da considerare per classe (vengono prese in considerazione gli oggetti da indice 0 a n-1).
- *nSeqxObj*: Numero di sequenze da creare per ciascun oggetto.
- *elevationProb*: Probabilità che la variazione tra un frame e l'altro riguardi l'elevazione (un incremento o decremento della dell'elevazione ha uguale probabilità).
- *azimuthProb*: Probabilità che la variazione tra un frame e l'altro riguardi l'azimuth (un incremento o decremento della dell'azimuth ha uguale probabilità).
- *lightingProb*: Probabilità che la variazione tra un frame e l'altro riguardi la luminosità (un incremento o decremento della della luminosità ha uguale probabilità).
- *flipProb*: Probabilità di cambiare direzione di variazione.
- *seqLen*: Lunghezza di ciascuna sequenza.
- *seed*: Seme per il generatore di numeri pseudo-casuali.
- *min distance*: Minima distanza tra ciascun frame del training set e del test set rispetto a ciascun oggetto.

In Fig.5.1 è possibile osservare un esempio di intestazione del file di configurazione generato a seguito dell'esecuzione dello script *NorbSamplerTest.java*. In seguito, precedute da una linea vuota e da un'intestazione specifica della sequenza vi sono tutte le immagini costituenti la stessa, identificate dal loro nome. Ogni sequenza ha un'intestazione contenente gli indici rispettivi alla classe, all' oggetto ed alla sequenza come è possibile osservare in Fig. 5.1. Dunque, riguardo il formato, non vi sono differenze rispetto alle sequenze generate nel training set.

```
Config Type: Test
-------------------------
nClass: 2
nObjxClass: 10
nSeqxObj: 1
ElevationProb: 0.3
AzimuthProb: 0.3
LightingProb: 0.2
FlipProb: 0.2
seqLen: 20
seed: 1234
minDistance: 3
-------------------------
```

Figure 5.1: Un esempio di intestazione del file di configurazione per il test set.

```
---------------
Class:    0
Object:   0
Sequence: 0
---------------
00_00_16_03.bmp
00_00_18_03.bmp
00_01_18_03.bmp
00_01_18_02.bmp
00_02_18_02.bmp
00_03_18_02.bmp
00_02_18_02.bmp
...
```

Figure 5.2: Un esempio di sequenza con relativa intestazione.

6 NORBSEQEXPLORER.JAVA

NorbSeqExplorer.java è un browser di immagini che si propone di offrire un interfaccia agevole per la visualizzazione delle sequenze create mediante gli script *NorbSamplerTrain.java* e

NorbSamplerTest.java. In Fig 6 si riporta un esempio di utilizzo. Mediante l'interfaccia è possibile specificare il file di configurazione da utilizzare (che sia di train o di test) e la directory principale nella quale sono contenute tutte le immagini separate per classe. Fatto ciò è possibile navigare le sequenze modificando i campi di input testuali e premendo invio oppure pigiando i bottoni laterali "prev" e "next". Si evidenzia che il codice in questione è ancora in uno stadio prematuro e non contiene adeguati controlli di sicurezza. Ad esempio, non offre ancora la possibilità di cambiare file di configurazione una volta selezionatone uno.

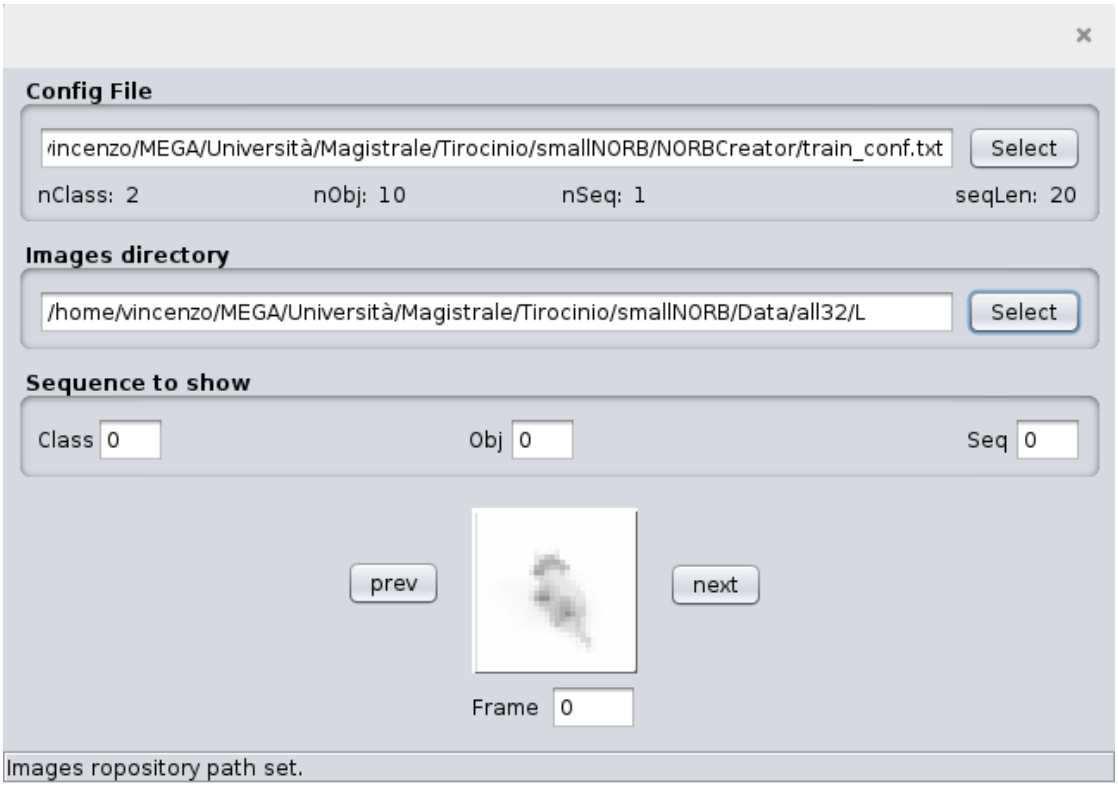

Figure 6.1: Interfaccia dell'esploratore di sequenze

7 NORBKNN.JAVA

NorbKNN.java è la classe di base che implementa un classificatore di sequenze mediante l'algoritmo KNN. In particolare si utilizza l'implementazione offerta dalla libreria *weka*. Per classificare un singolo frame della sequenza di test è inoltre possibile scegliere se fondondere o no i livelli di confidenza rispetto ai frame precedentemente classificati.

8 KNNTestSeries.java

KNNTestSeries.java è una classe per il testing automatico dell'accuratezza al variare di numerosi parametri come la lunghezza delle sequenze o la distanza del test-set dal training-set.

9 SeqVerifier.java

SeqVerifier.java è una classe per la verifica automatica delle sequenze generate. In particolare si preoccupa di verificare se le sequenze sono effettivamente sequenziali e se le distanze dal training set sono veritiere. Per ogni sequenza generata, la verifica è effettuata automaticamente, quindi è impossibile produrre sperimentazioni su sequenze malformate.

10 NorbCreator.java

NorbCreator.java si preoccupa di offrire un' interfaccia testuale per l'automatizzazione di tutte le funzionalità offerte. Mediante la somministrazione da parte dell'utente di un unico file di configurazione completo di tutte le informazioni, infatti, è possibile passare direttamente alla visualizzazione delle sequenze create.

10.1 Il formato del file di configurazione

Il formato del file di configurazione è autoesplicativo e rappresenta l'unione di tutte le informazioni necessarie per svolgere la totalità delle operazioni descritte precedentemente. Si consideri la Fig. 10.1 come esempio.

```
#########################
#      CONFIG FILE      #
#########################

#########################
#   CONVERSION PARAMS   #
#########################
matlabFile: ../Data/Matlab/smallnorb-5x46789x9x18x6x2x96x96-training-
destDir: ../Data/all32
convert(yes/no): no
inputWidth: 96
inputHeight: 96
scaleFactor: 3
#########################

#########################
#     COMMON PARAMS     #
#########################
imagesRepo: ../Data/all32/L
nClass: 2
nObjxClass: 10
elevationProb: 0.55
azimuthProb: 0.35
lightingProb: 0.1
flipProb: 0.05
seqLen: 40
#########################

#########################
#     TRAIN PARAMS      #
#########################
fileName: train_conf.txt
nSeqxObj: 1
seed: 1
#########################

#########################
#      TEST PARAMS      #
#########################
fileName: test_conf.txt
nSeqxObj: 1
seed: 2
minDistance: 0
#########################

#########################
#      END CONFIG       #
#########################
```


Figure 10.1: Un esempio completo di file di configurazione.

11 NORBCREATOR.JAR

NorbCreator.jar è un jar eseguibile contenente tutte le classi descritte precedentemente e le librerie indispensabili all'esecuzione. Esso, quando invocato, esegue il main della classe di interfaccia *NorbCreator.java* passandole gli opportuni parametri inseriti da line di comando. In particolare è possibile eseguire il jar secondo le seguenti modalità:

```
1) java -jar norbCreator.jar configFileName
2) java -jar norbCreator.jar --convert configFileName
3) java -jar norbCreator.jar --sampleTrain configFileName
4) java -jar norbCreator.jar --sampleTest configFileName
5) java -jar norbCreator.jar --seqExplorer configFileName
```

Passando semplicemente il nome del file di configurazione tutte le sezioni vengono considerate e le conseguenti operazioni eseguite. Quindi, senza modificare alcuna linea di codice, è possibile tarare direttamente i paramentri nel file di configurazione, lanciare il programma e visualizzare le sequenze create non appena il browser farà la sua comparsa sullo schermo. Utilizzando invece gli opportuni flag solo le corrispondenti operazioni vengono svolte in maniera esclusiva.