Atelier OCR et HTR

Jean-Baptiste Camps

Masters HNC et TNAH | ENC (PSL)

7 janvier 2019

Acquisition du texte

Dans la constitution d'un corpus de textes, la première phase est bien sûr l'acquisition du contenu des textes envisagés.

Transcription des témoins, selon les critères scientifiques du projet (transcriptions allographétiques, graphématiques, normalisées ; édition critique ; etc.). Méthode souvent la plus sûre, mais aussi la plus lente ;

"Transcription" assistée par ordinateur en utilisant un algorithme permettant la reconnaissance optique de caractères (*optical character recognition* ou OCR) imprimés, ou la reconnaissance des écritures manuscrites (*handwritten text recognition* ou HTR).

Téléchargement de textes depuis des corpus en lignes, des sites d'édition électronique, des bases de données d'éditeurs, etc.

OCR et HTR

Reconnaissance optique des caractères (imprimés)
Optical character recognition (OCR)

- "problème résolu" de l'informatique ;
- aisé d'obtenir des taux d'erreur caractère (CER) $< 2\%$;
- outils libres : `Tesseract 4`, … ;
- existence de modèles génériques (par langue).

Reconnaissance des écritures manuscrites
Handwritten text recognition (HTR)

- très peu fonctionnel jusqu'à ces dernières années ;
- nouveaux développements : IA (réseaux de neurone récurrents LSTM…) ;
- outils libres : `OCRopy`, …
- modèles spécifiques à entraîner (pour chaque main, écriture,…).

Les étapes

1. traitement des images ;
2. analyse de la mise en page et identification des lignes ;
3. reconnaissance des caractères ;
4. d'éventuels post-traitements, visant à améliorer les résultats.

Dans une démarche de reconnaissance des écritures, la qualité des images et de leurs traitements est cruciale.

Besoins :

1. images en 300 DPI ;
2. redressées, débruitées ;
3. binarisées.

Outils : logiciels de traitement d'image, par ex. `ScanTailor`

T.P. ScanTailor

1. Démarrer un nouveau projet ;
2. charger les images du dossier `digby_23` ;
3. suivre les différentes étapes dans le logiciel ;
4. exporter en tiff binarisé 300 DPI.

Plan

Analyser la mise en page

Identifier

- zones de texte ;
- décoration ;
- colonnes ;
- lignes ;
- mots ;
- lettres.

Approches

- Sans apprentissage, par ex.
  - `OCRopy 1` ;
  - Oriflamms (IRHT) ;
- fondée sur des méthodes d'apprentissage (IA), par ex.
  - `OCRopy 2` ?

Installer Kraken

Une installation de python ≥ 3.6 est nécessaire.

Sur Ubuntu, il vous faut les paquets `python3.6`, `python3.6-dev` et `pip3`.

```
# Créer un environnement virtuel (optionnel)
$ virtualenv env -p /usr/bin/python3.6
# l'activer (optionnel)
$ source env/bin/activate
# installer
$ pip3 install kraken
# Utilisation basique
# installer le modèle par défaut
$ kraken get default
# lister les modèles disponibles au téléchargement
$ kraken list
```

Analyser la mise en page avec Kraken : étape par étape

Depuis la racine du dossier :

Si les images n'ont pas été prétraitées

```
#Binariser les images (si pas déjà fait
#  avec ScanTailor)
$ kraken -I "src_digby23/*" -o .png binarize
# Segmentation en lignes
$ kraken -I "src_digby23/*.png" -o .json segment
# OU les deux d'un coup
kraken -I "src_digby23/*" -o .json binarize segment
```

Si elles l'ont été

```
# Segmentation en lignes
$ kraken -I "tif/*" -o .json segment
```

Tout-en-un

Pour binariser, segmenter et générer un fichier de transcription

```
# Pour générer un fichier de transcription
$ ketos transcribe -o gt.html tif/*
```

Différentes solutions techniques

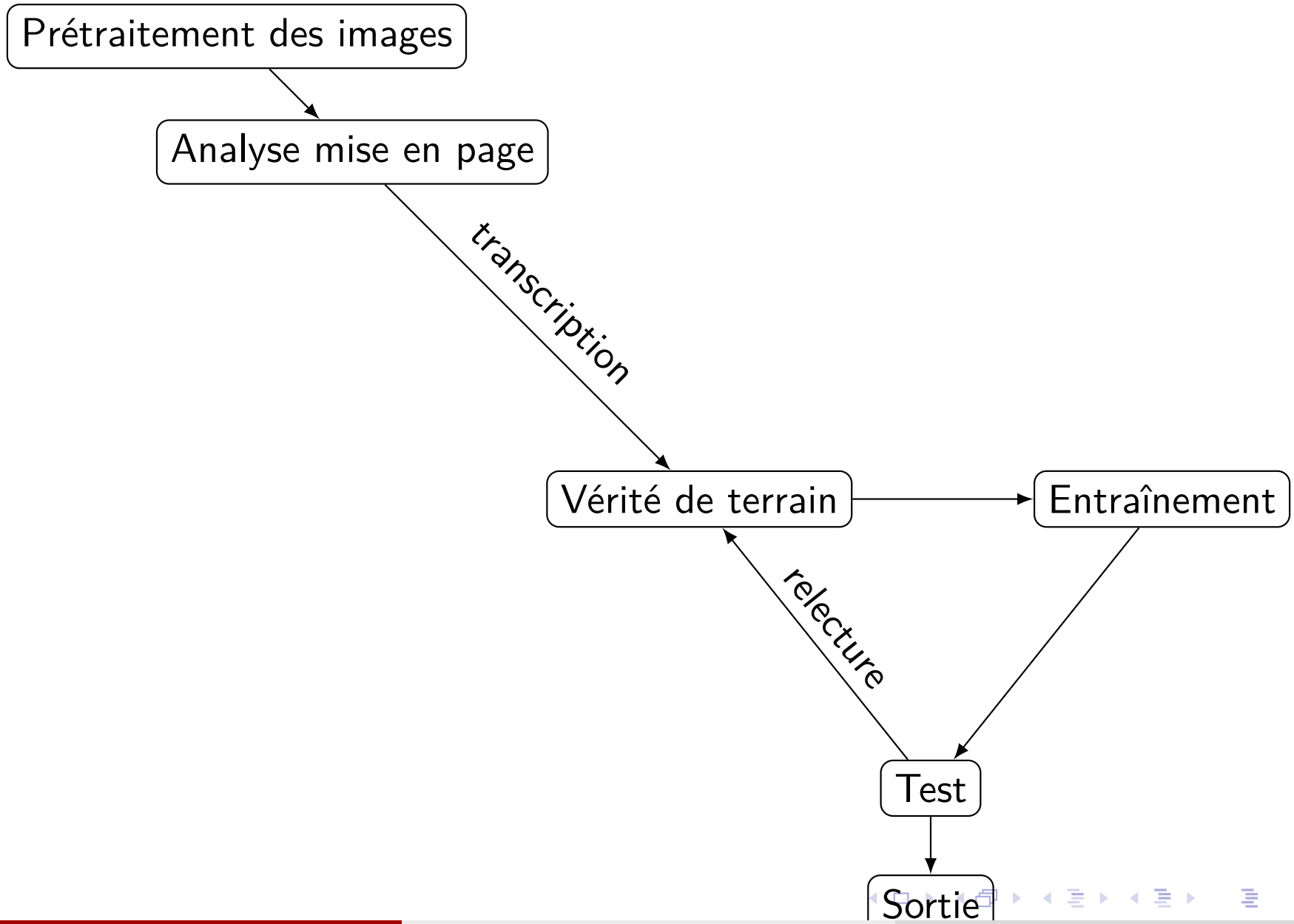
- approches segmentées ou non segmentées ;
- mesures de distance ; méthodes statistiques (chaînes de Markov) ou d'intelligence artificielle (réseaux de neurones convolutifs ou récurrents, LSTM 1D, LSTM 2D, etc.) ;
- outils directement opérationnels ou nécessitant un entraînement.

Ocropy, CLSTM et Kraken

`OCRopy` et CLSTM développés par Thomas M. Breuel ; `Kraken`, *fork* d'`OCRopy` développé par Ben Kiessling (PSL).

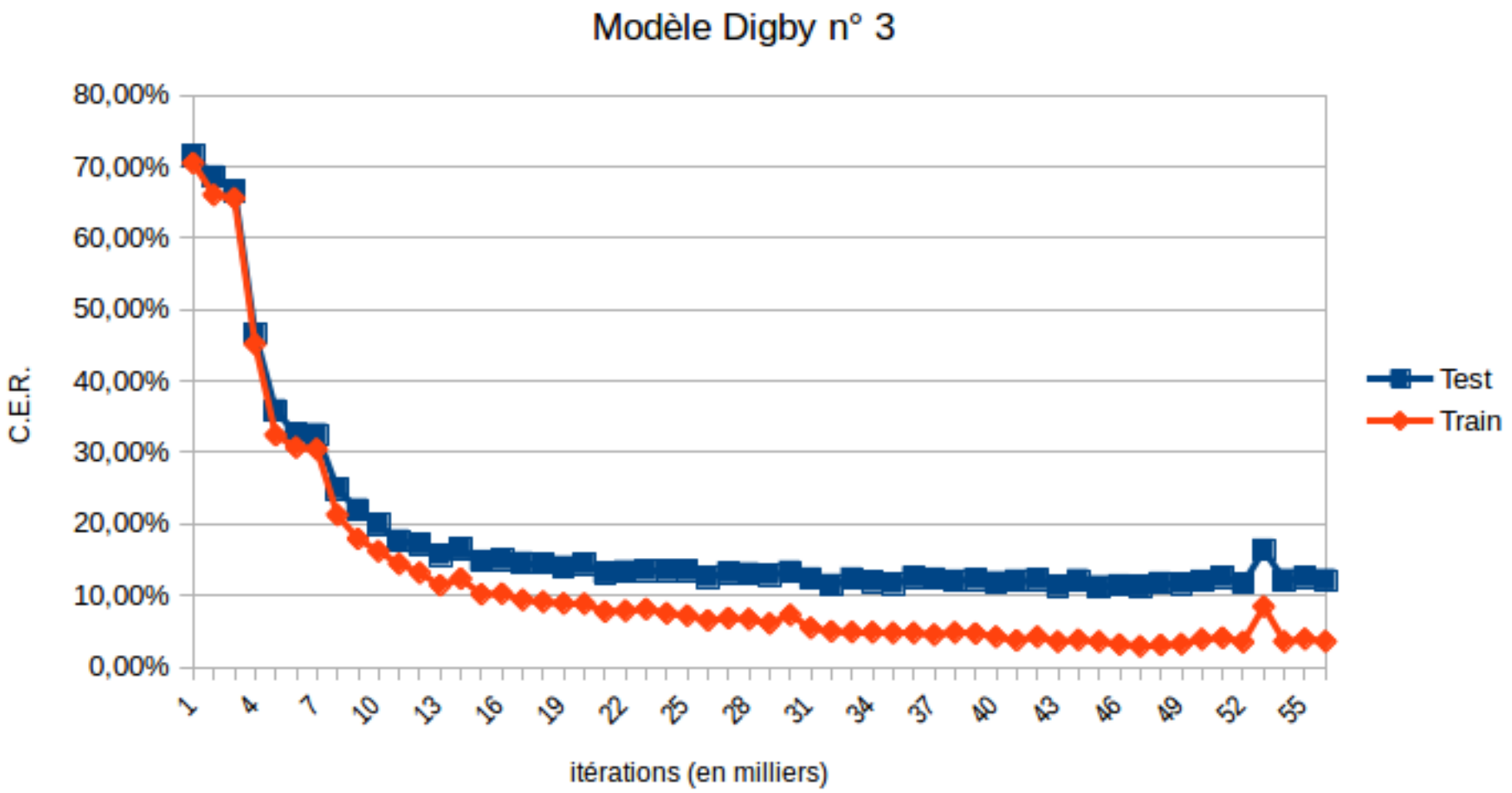
- approche non segmentées ;
- réseaux de neurones récurrents (LSTM) ;
- *open source* et nécessitant l'entraînement d'un modèle.

Un apprentissage guidé



Un apprentissage guidé

Entraînement sur un ms. (ici Roland d'Oxford)



Un apprentissage guidé

Résultats

- pour un imprimé ancien, des taux d'erreur de l'ordre de 1% sont atteignables...
- pour un ms., des taux inférieurs à 10% sont atteignables ;
- cas du Digby 23, env. 400 lignes d'entraînement, taux de succès de 89,16% en test, et 93% sur l'ensemble des données.

Un apprentissage guidé

Évaluation et confusions fréquentes

```
Evaluating model_best.mlmodel
Evaluating  100\%
=== report  ===
21270 Characters
1473 Errors
93.07\% Accuracy

1152 Insertions
93 Deletions
228 Substitutions

Errors Correct-Generated
557 { 0xa } - {  }
85 { SPACE } - {  }
41 { ı } - {  }
38 {  } - { SPACE }
30 { n } - {  }
```

Plan

T.P. Kraken : Préparation des données

1. Essayer d'appliquer un modèle déjà entraîné

```
# Lancer la reconnaissance
# et extraire un fichier de correction
$ ketos transcribe -o gt.html --prefill model_best.mlmodel
↪  tif/*.tif
# Ou, si ça vous satisfait, extraire directement un fichier texte
$ kraken -I "tif/*.tif" -o .txt segment ocr -m model_best.mlmodel
# ou hocr
$ kraken -I "tif/*.tif" -o .txt segment ocr -m model_best.mlmodel
```

Pas terrible... Mais comment améliorer le modèle ?

2. Corriger / transcrire

Plan

T.P. Kraken : préparer l'entraînement

3. Extraire et entraîner

```
#Extraire et normaliser les caractères
$ ketos extract --output book --normalization NFD gt.html
```

Ensuite 90% des lignes corrigées vont servir à l'entraînement et 10% au test des modèles.

- il est recommandé de faire cette répartition de manière aléatoire ;
- il est possible d'accroître artificiellement le nombre de lignes d'entraînement, en bruitant de différentes manières les lignes de GT.

Par défaut, Kraken réalisera certaines de ces opérations pour vous (contrairement à OCRopy), mais il est aussi possible d'utiliser un logiciel comme Doccreator, par exemple.

T.P. Kraken : lancer l'entraînement

On peut ensuite lancer un entraînement (de zéro ou à partir du modèle précédent, aux choix).

```
#Lancer l'entraînement
$ ketos train book/*.png
```

N.B : de nombreux autres paramètres sont disponibles, liés au modèle et à l'entraînement.
Si les données n'ont pas été normalisées auparavant, on peut utiliser l'option '-u NFD' (normalisation Unicode).

Plan

T.P. Kraken : Test des résultats

4. Tester les résultats

Une fois que l'entraînement a atteint un niveau satisfaisant, on peut tester la qualité du résultat,

```
# Test du meilleur modèle et confusions de caractères
$ ketos test -m model_best.mlmodel book2/*/*.png
```