

# Module 0

## Software needed for the OCR workshop

Uwe Springmann

Centrum für Informations- und Sprachverarbeitung (CIS)  
Ludwig-Maximilians-Universität München (LMU)



2015-09-11

# Introduction

This is a list of software requirements to successfully complete the practice sessions and maximize your learning experience of the workshop on *OCR and postcorrection of early printings for digital humanities*.

- on OS: Linux is your best option; MacOS can be used almost as well (some problems might arise with OCRopus installation); Windows users will not be able to train or run OCRopus models (they may consider a virtualbox Linux installation)
- our focus is on open source software
- for Linux, almost all software is available in its package repositories and can be installed from there
- you will be optimally prepared if the software is running on your laptop before the workshop begins, and you know how to use it
- the software packages mentioned below are part of a complete OCR toolchain - if you miss some parts because of installation problems, we will provide you with suitable input data for each separate step

# OCR engines I

- we will treat *Tesseract* and *OCROpus* (open source engines) as well as *Abbyy Finereader* (commercial)
- Tesseract and OCROpus can be downloaded and installed locally; Abbyy will provide a demo key for its online service
- both Tesseract and OCROpus have recently been moved from Google Code to GitHub (click on blue inline links and follow the installation procedures):
  - [Tesseract](#): available for Linux, Mac, Windows
  - [OCROpus](#): Linux (Mac)
  - OCROpus (now called Ocropy) can be installed in your home directory:

```
python setup.py install --home=~/<install-dir>
export PATH=$HOME/<install-dir>/bin:$PATH
export PYTHONPATH=$HOME/<install-dir>/lib/python
```

- if you want to install OCROpus in a docker environment (Mac):
  - use the [Ocrocis](#) wrapper

## OCR engines II

- don't forget to also download some language specific Tesseract training files
  - deu\_frak.traineddata: German Fraktur
  - grc.traineddata: Ancient Greek from [Nick White](#)
  - lat.traineddata: Classical Latin from [Ryan Baumann](#)

## Graphical frontends for Tesseract (optional)

- **gImageReader**: Linux & Windows
  - **Windows installation tips**
- **VietOCR**: Linux, Mac, Windows

## Downloading and installing the postcorrection tool PoCoTo

- download the [binary distribution of PoCoTo](#)
- this will download a zipped archive file `ocrcorrection.zip`.
- extract (unzip) this archive to a convenient place somewhere in your user directory
- this will create a folder `ocrcorrection`
- in the folder `ocrcorrection/bin`, identify the appropriate executable for your operating system:
  - MS Windows: either `ocrcorrection.exe` or `ocrcorrection64.exe`
  - otherwise it is the file `ocrcorrection`
- before you start the application, make sure that the [Java Runtime Environment \(jre\)](#) is installed on your system
- PoCoTO is described in detail in its [manual](#) which may be consulted for any questions

# Preprocessing tools

- split pdf into single page images:
  - pdftk from [PDFtk](#)
- further pdf tools:
  - pdftoppm, pdfimages from [Xpdf](#) tools, or (Linux) from poppler-utils package
- format conversion:
  - convert from [ImageMagick](#)
- further preprocessing: [ScanTailor](#)
- [learn how to use ScanTailor](#)

## OCR evaluation toolkit

- we need to be able to evaluate OCR output against ground truth
- a widely used tool collection is the Rice/Nartker UNLV/ISRI OCR evaluation toolkit
- [user guide and source code with UTF-8 wrapper from Nick White](#)
- needs to be compiled locally



# Fonts with good glyph coverage

- Fonts supporting old glyphs
- Junicode is used for this workshop