# DELIVERABLE

| | |
|---|---|
| **Project Acronym:** | Europeana Newspapers |
| **Grant Agreement number:** | 297380 |
| **Project Title:** | A Gateway to European Newspapers Online |

# D3.6 Planning resources and quality estimation toolkit

| | |
|---|---|
| **Revision:** | 1.0 |
| **Authors:** | **Christian Clausner, USAL**<br>**Apostolos Antonacopoulos, USAL** |

| **Project co-funded by the European Commission within the ICT Policy Support Programme** | | |
|---|---|---|
| **Dissemination Level** | | |
| **PU** | **Public** | **x** |
| **PP** | **Restricted to other programme participants (including Commission Services)** | |
| **RE** | **Restricted to a group specified by the consortium (including the Commission Services)** | |
| **CO** | **Confidential, only for members of the consortium and the Commission Services** | |

## Revision History

| Revision | Date | Author | Organisation | Description |
|---|---|---|---|---|
| 0.1 | 27-02-2015 | Christian Clausner | USAL | First draft |
| 0.2 | 25-03-2015 | Apostolos Antonacopoulos | USAL | Final draft |
| 0.3 | 27-03-2015 | Günther Hackl | UIBK | Internal review |
| 0.4 | 02-04-2014 | Apostolos Antonacopoulos | USAL | Final draft after internal review |
| 1.0 | 27-04-2015 | Clemens Neudecker & Sandra Kobel | SBB | Internal review and final version |

## Contents

## Executive Summary

This report presents a workflow for the evaluation of pilot projects, which can enable the planning of resources and decision making for potential digitisation projects. It combines earlier findings mainly from Tasks 3.1 (use scenarios), 3.2 (evaluation datasets), 3.3 (evaluation tools), and 3.5 (performance evaluation).

In addition, an approach (complete with software tools) for estimating the performance of digitisation pipelines is introduced. As experimental evidence shows, the quality of digitisation results can be predicted reasonably accurately, using concepts from machine learning, based on only the scanned pages (with minimal training). Quality estimation can complement pilot projects and can be used for purposes of triage (selection of documents to be included in digitisation projects).

# 1 Introduction

Part one of this report (Section 2) presents the evaluation workflow created for pilot projects and the relevant software tools to realise it, available through the University of Salford (USAL) internet presence:
http://www.primaresearch.org/tools.

Part two (Section 3) describes a quality estimation workflow and introduces two newly developed tools for feature extraction, as well as another new software tool called "Quality Estimation", available here:
www.prima.cse.salford.ac.uk/tmp/ENP/FeatureExtractionAndQualityEstimationTools.zip.

# 2 Performance Evaluation for Pilot Projects

Resources for digitisation of printed material are limited and can only cover a fraction of the complete holdings of libraries and archives. It is therefore crucial to be able to make **informed decisions** on which selections of holdings to digitise and which to omit (for the short/medium term at least). In addition, it is also important to be able to assess whether some of the existing digitised material is worth rescanning and reprocessing with the current state-of-the-art technologies (to significantly improving image and resulting full-text quality).

Given a selected subset of documents and an intended eventual use-scenario, a small-scale pilot project using only a few documents, can help to determine if a large-scale digitisation endeavour is feasible and at what cost. In the following we describe the general workflow on how to realise a pilot project (Section 2.1) and what resources are available (software tools and data repositories) – in Section 2.2. An example using the Europeana Newspapers Ground Truth set, in Section 2.3, completes this part of the report.

## 2.1 Evaluation Workflow

A pilot project comprises certain steps in order to evaluate if a selected collection of printed material is suitable for digitisation, with a specific use scenario in mind. This can be described as a workflow including (see also Figure 1):

- Careful selection of a (small) dataset
- Processing the data with the target digitisation pipeline
- Creating ground truth (to be used as reference data)
- Measuring the quality of the output of the digitisation
- Making decisions based on the measured performance

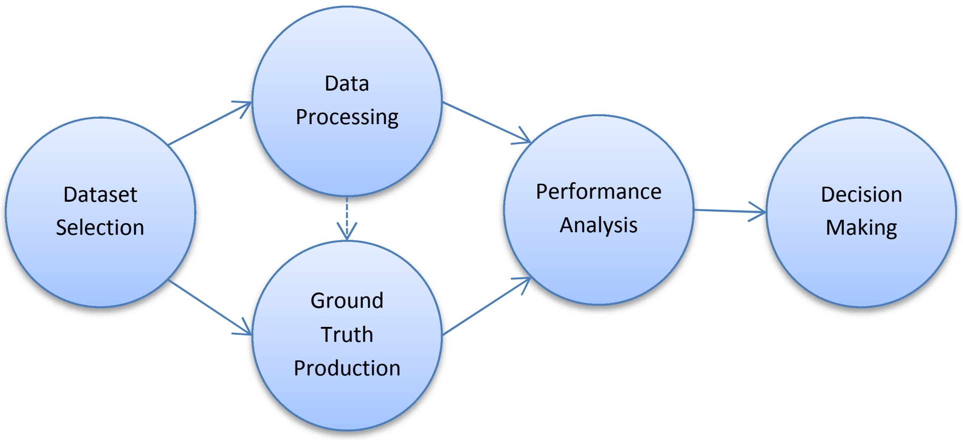


**Figure 1 – Evaluation workflow**

Each step is detailed in the following subsections.

### 2.1.1 Dataset Selection

The selection of a dataset for a pilot project is usually driven by two major constraints:

1. To narrow down the number of documents/pages so as to be in line with the available resources (pilot budget).
2. To maintain the representativeness of the dataset with respect to the full collection as far as possible.

The size of the dataset should be fixed to a size which allows for reasonable variety while keeping costs within the limits of the budget for the pilot. Costs per page for scanning and Ground Truthing vary considerably depending on the type of document (size, language, etc.). For the Europeana Newspapers dataset, for example, the quoted costs (Ground Truthing only) ranged from 1.25 to 2.82 EUR per 1000 characters.

With regard to representativeness, the goal should be to maintain the distribution of languages, scripts, title pages, middle pages, and characteristic layouts as close to the full collection as possible. For practical reasons and to be able to run realistic evaluation scenarios it can be of advantage to include at least one complete document (e.g. a newspaper issue), if applicable.

### 2.1.2 Data Processing and Ground Truth Creation

To obtain meaningful results, the selected dataset should be processed using a digitisation pipeline that is identical or at least as close as possible to the eventual target pipeline, which is intended to be used for the complete collection (if the outcome pilot project suggests a "go ahead"). This can also include multiple iterations using variations of the pipeline, if a stable approach (specific pipeline) has not been established yet. In addition to proprietary digitisation pipelines, the data can also be processed using state-of-the-art open source software, if desired. This can help to establish a baseline for comparison.

Ground Truth, representing the ‘perfect’ output of a digitisation method, is required as a point of reference. Quality measurements are made with regard to the Ground Truth data. It should be noted, however, that while 100% accuracy is desirable for Ground Truth, this is usually cost-prohibitive (or practically impossible) for such a manually-assisted process. For the Europeana Newspapers project Ground Truth production was aimed at 99.95% accuracy. Automated validation tools can help to arrive at the desired outcome.

Depending on the chosen digitisation pipeline and the target use scenario(s), different aspects of a page need to be Ground Truthed, for instance:

- Precise region (e.g. paragraph, illustration etc.) outlines
- Region type labels (e.g. “text – paragraph” or “image”)
- Full text (Unicode encoded, including special characters such as symbols and ligatures)
- Reading order

In some cases it can be of advantage to pre-produce (partially complete) Ground Truth data using an automated method. To obtain the preferred accuracy, those preliminary results have to be corrected manually using suitable software tools.

The data formats that are being used and the consistency of the data are also important for a successful evaluation. Performance analysis methods require method outputs and Ground Truth to be stored in the same data format. This can be achieved by producing the data in the desired format or by converting it later on. Similar considerations need to be made towards data consistency. This usually involves conventions for certain aspects of the data. Text content, for instance, can contain ligatures in form of single characters (e.g. “Æ”) or in their expanded version (e.g. “AE”). Consistency can be achieved by preceding the performance analysis with suitable normalisation steps.

### 2.1.3 Performance Analysis

Choosing the most appropriate approach for measuring the quality of the digitisation results depends heavily on the use-scenario that the data is eventually intended for. Within the Europeana Newspapers project, questionnaires were used to identify real-world scenarios (see *D3.1 Evaluation profiles for use scenarios*), of which the most relevant were translated into following evaluation profiles (aggregation of evaluation methods, settings, and weights):

- Keyword Search in Full Text
- Phrase Search in Full Text
- Access via Content Structure
- Print/eBook on Demand
- Content-Based Image Retrieval

Furthermore it can be taken into consideration how detailed the performance analysis should be. Simple *benchmarking*, for instance, delivers basic figures that can be used for comparing different digitisation approaches. In-depth *analysis*, on the other hand, can give useful insights to where the weak points of a digitisation pipeline are, enabling future improvements.

Metrics can be calculated for every individual step of a digitisation pipeline, including (but not limited to):

- Page segmentation
- Region classification
- Reading order detection
- Text recognition (OCR)

See deliverable *D3.1 Evaluation profiles for use scenarios* for more information.

#### 2.1.4 Making Decisions

Based on the results of the performance analysis, it can be decided:

- To go ahead with the digitisation of the complete collection
- To choose a subset of the most promising documents/pages for digitisation
- To enhance the digitisation pipeline before proceeding
- To use a combination of the above
- To disregard the full collection for short and medium-term digitisation

A prescriptive guide for how to make this decision cannot be provided. If, for example, the target scenario is keyword-based text search, a text recognition accuracy of 80% might be considered sufficient. In other scenarios a much higher quality might be necessary. In practice, experts need to study the evaluation results in detail, using appropriate tools for visualisation and calculation of trends, etc. Comparison against state-of-the-art methods can be beneficial in this context.

## 2.2 Resources

Apart from valuable expertise gained, a comprehensive set of tools and datasets has been produced within EU-funded projects such as IMPACT[1], SUCCEED[2], and Europeana Newspapers itself. In the next two subsections we describe the most prominent of those resources.

### 2.2.1 Datasets

Where the execution of a comprehensive pilot project including scanning and Ground Truth production is impossible, an evaluation based on existing data can be considered. Large datasets including Ground Truth are available. Powerful search mechanisms of the online data repositories allow the selection of material within the dataset that is close to that of the target collection.

[1] http://www.impact-project.eu/

[2] http://www.succeed-project.eu/

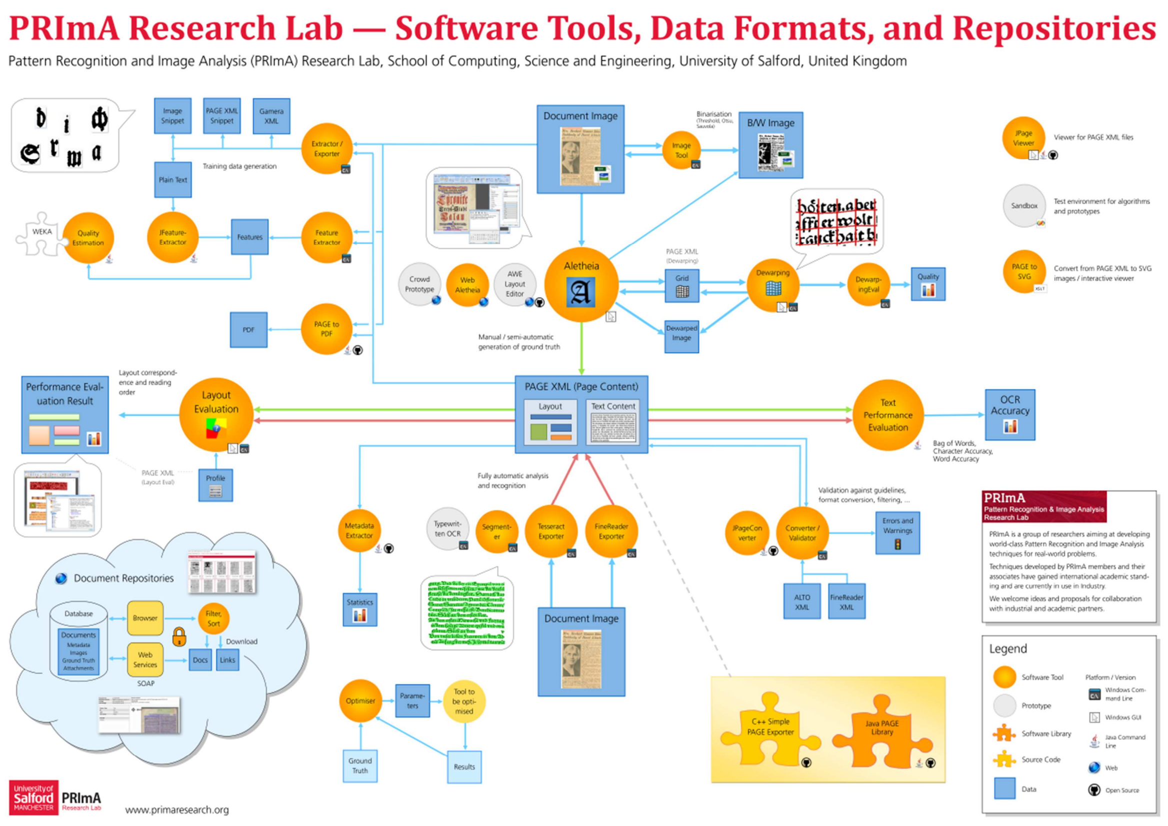
Deliverable *D3.2 Evaluation dataset including ground truth* provides a detailed description of the online repository used for Europeana Newspapers and the dataset itself. It contains 600 newspaper page images with Ground Truth, representing a representatively wide range of languages, publication periods, and conditions of the original printed material. In addition to common metadata, all pages have also been tagged using over 80 keywords representing various features, issues, and artefacts, from categories such as: Page content, layout, production, faults, wear, aging and use, distortions and noise from digitisation.

The datasets from IMPACT (continued by SUCCEED) and Europeana Newspapers are hosted at USAL: http://www.primaresearch.org/datasets

### 2.2.2 Tools

Deliverable *D3.3 Evaluation Tools – final versions* provides an extensive report on available evaluation tools and data formats. Access is managed through USAL's web presence (http://www.primaresearch.org/tools) and in some cases also including source code on GitHub (https://github.com/PRImA-Research-Lab). See Figure 2 for a snapshot of the infrastructure of tools, libraries, and data formats developed by USAL, the most essential being:

- The PAGE (Page Analysis and Ground truth Elements) data format for page layout and text content,
- The Aletheia Ground-Truthing system and result viewer,
- The Layout Evaluation tool for page analysis results, and
- The Text Evaluation tool for OCR (text only) results



PRImA is a group of researchers aiming at developing world-class Pattern Recognition and Image Analysis techniques for real-world problems.

Techniques developed by PRImA members and their associates have gained international academic standing and are currently in use in Industry.

We welcome ideas and proposals for collaboration with industrial and academic partners.

**Figure 2 – Map of tool and format infrastructure by USAL**

## 2.3 Example

The Europeana Newspapers dataset of 600 newspaper pages is an example case of a pilot project. The images are representative of the holdings of 12 project partners (50 pages each). Figure 3 shows the implementation of the workflow from Figure 1. Two different digitisation pipelines using ABBYY FineReader Engine 11 have been used. For comparison, also the open source OCR engine Tesseract 3.02 has been applied as a third pipeline. The Ground Truth was partially pre-produced using FineReader Engine 10 and corrected by service providers. The quality of the Ground Truth was checked both by human operators and using the automated PAGE Validator tool. Where necessary, results from OCR systems were converted to PAGE format using USAL's PAGE converter. The same tool was also used for normalisation (in this case to apply text filters). The performance of the OCR systems was measured using the USAL Layout Evaluation tool and the USAL Text Evaluation tool. Finally, all results were accumulated and visualised in Excel spreadsheets. Deliverable *D3.5 Performance Evaluation Report* provides an in-depth report on the evaluation.

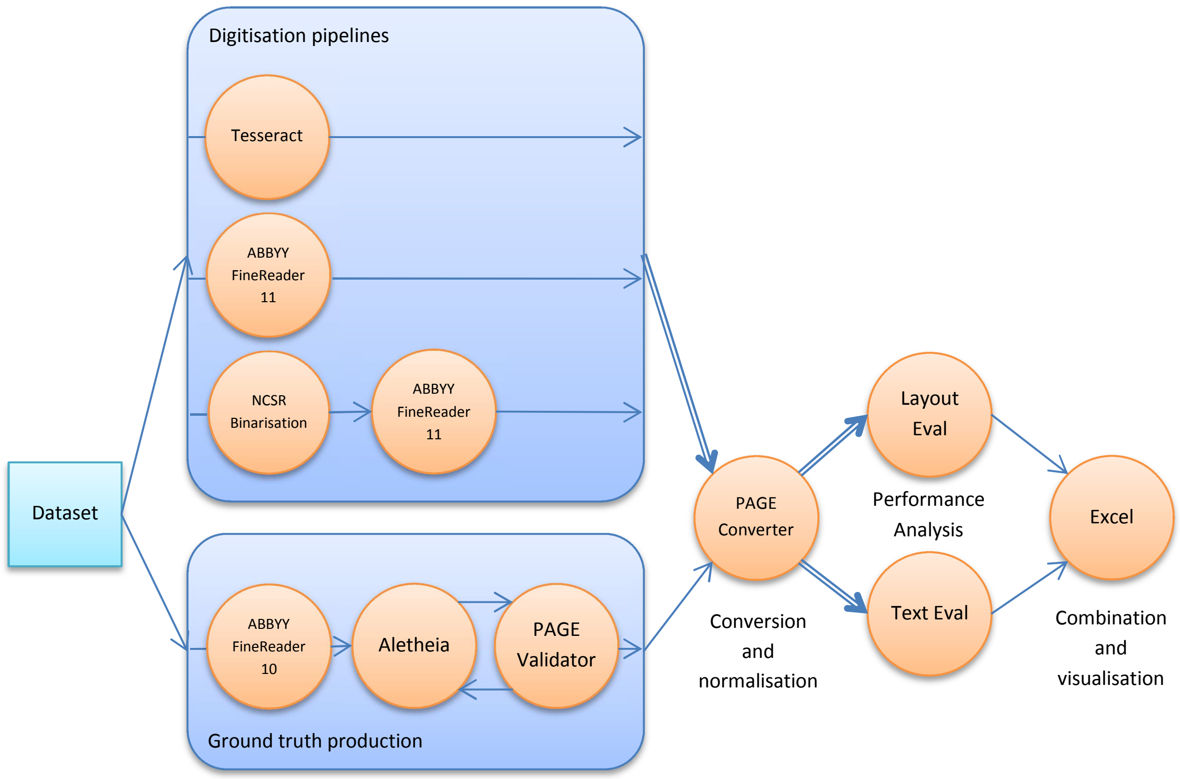


**Figure 3 – Evaluation workflow for ENP**

Figure 4 shows selected evaluation results for the three pipelines for each of the use-scenarios defined in *D3.4 Report on usability and potential of existing material*. A decision on the viability of a future digitisation project (for a specific use scenario) could now be made, based on the measured performance. Breaking down the success rates by language or other properties can further simplify the decision process (see for example Figure 5). Furthermore, additional in-depth performance information (see Figure 6) will help in guiding future efforts in improving the pipelines.

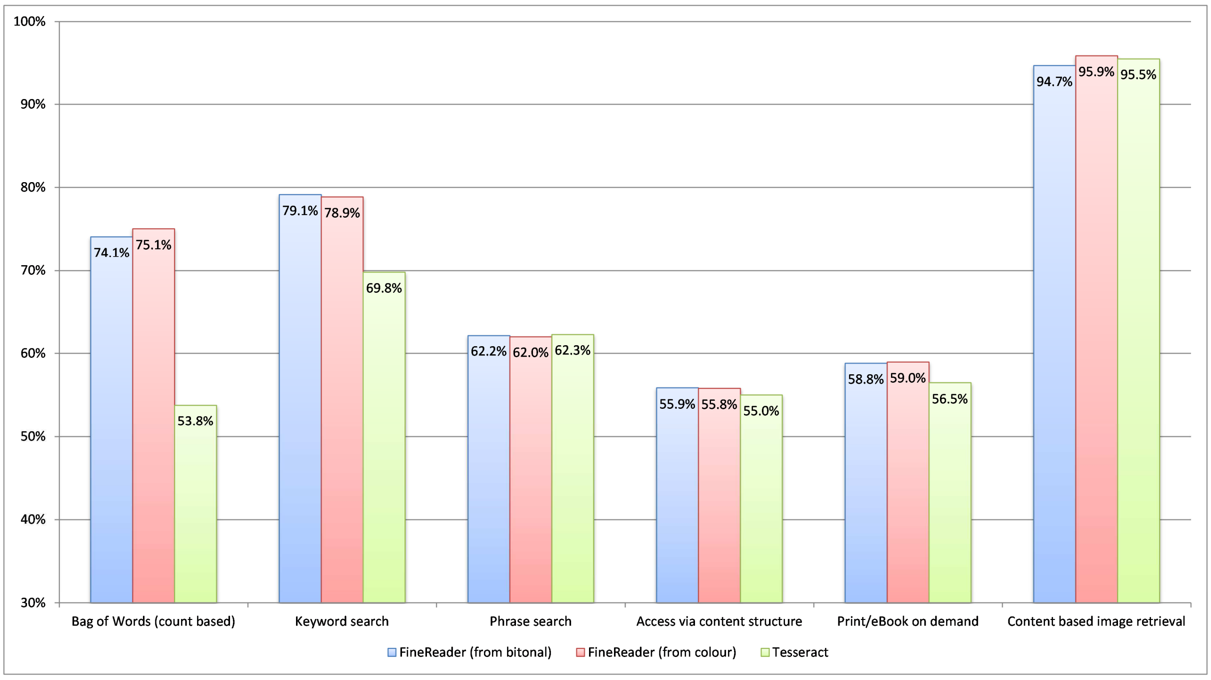


**Figure 4 – Evaluation results for three digitisation pipelines**

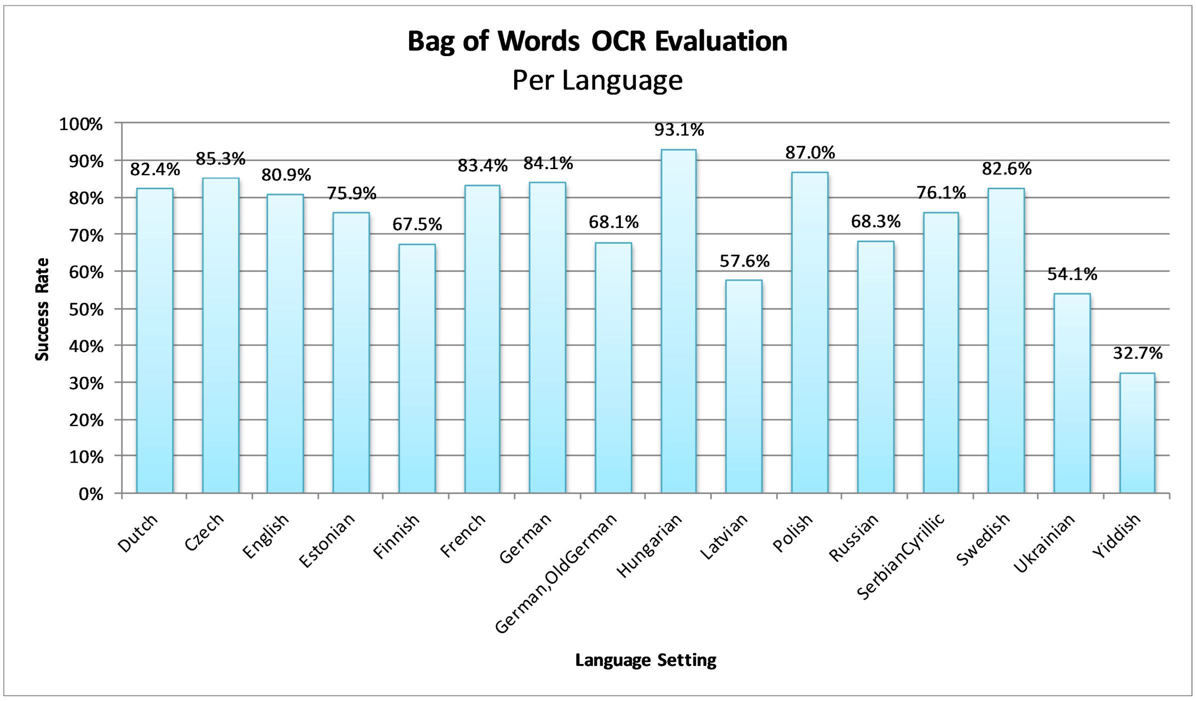


**Figure 5 - Bag of Words evaluation – per language (FineReader, bitonal images)**

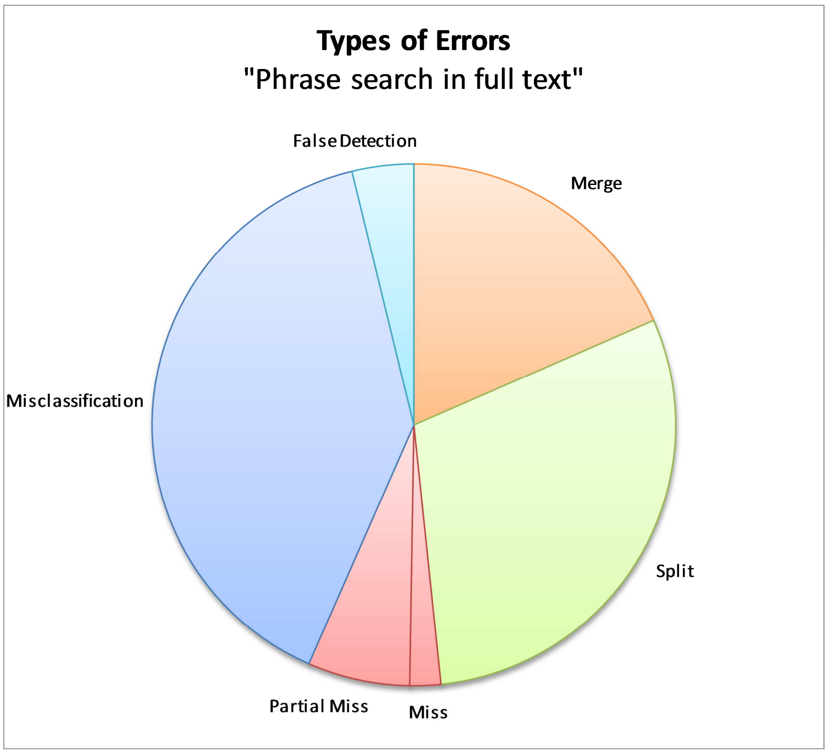


**Figure 6 - Proportions of layout analysis errors – Phrase search in full text (FineReader, bitonal images)**

Closely related to pilot projects are also international *competitions* that are usually organised under the auspices of established conferences such as the biannual ICDAR (International Conference of Document Analysis and Recognition). Examples are the ICDAR2013 Competition on Historical Book Recognition (HBR2013) and the ICDAR 2013 Competition on Historical Newspaper Layout Analysis (HNLA2013), organised by USAL. Participants such as research groups and digitisation service providers thereby compete in certain challenges against each other and against state-of-the-art systems. Similarities to pilot projects can be found in dataset selection, performance analysis, and interpretation of results.

# 3 Quality Estimation – Workflow and Experiments

Objectively measuring the success of an OCR pipeline requires precise Ground Truth, the creation of which (for a representative dataset) can involve considerable production costs. As an alternative, quality *estimation* can be used to approximate in advance the measure of the success of OCR if applied to the target collection. Ground Truth is still required, but for significantly fewer document pages.

## 3.1 Overview

The aim of the estimation is, based on just the scanned pages, to predict the quality of digitisation results that a given pipeline would produce. This can be achieved by using feature-based numeric prediction based on a classifier. The estimation workflow has two phases: (1) Training of a classifier and (2) Quality prediction using the classifier.

Classifier creation/training workflow:

1. Obtain ground truth for a small number (depending on the diversity of target collection) of document images
2. Extract features from the document images
3. Select best features and train classifier by comparing actual evaluation results with the estimated quality

Prediction workflow:

1. Extract features from the document images
2. Use the classifier to estimate a quality (e.g. OCR or segmentation success rate) from the features (numeric prediction)

The nature of the available features and the selection of the best features for classification are most crucial for the predictive strength of the quality estimation. The next two sections provide more details on these points.

## 3.2 Features

Features are numeric (e.g. 0.5, 0.3, 10) or nominal (e.g. “red”, “blue”, “yellow”) values that are either calculated or readily available (as metadata). For the purpose of classification, a feature ideally should have some correlation with the classification target (e.g. OCR quality). This is usually not easy to determine, sometimes only combinations of features produce a correlation. Therefore, the most common approach is to define a variety of features and use automated feature selection to find the strongest ones. A reduction of the number of features not only speeds up the classification but also improves the predictive quality in most cases.

While their combined use in the overall classification is the main purpose, some features can be interesting in themselves, providing more direct insights into the condition or potential quality of OCR results. The feature “Region Overlaps” (see 3.2.2) for instance, can hint at problems in the segmentation step of the OCR pipeline.

The next three subsections describe all the features that have been used in the quality estimation experiments for the Europeana Newspapers Ground Truth dataset.

### 3.2.1 Features from metadata

Some basic features are expected to be available as metadata that is stored together with the document image. For the conducted experiments these were:

- Language (e.g. English, German, OldGerman)
- Font (normal, gothic, mixed)

### 3.2.2 Image, page layout, and text features

A range of features (potentially relevant to quality prediction) based on document image, page layout, and detected text have been defined. The source for page layout representation and text content can be either the OCR system of the production workflow (whose quality is to be predicted) or, if this is not feasible, the open source OCR system Tesseract.

Two tools for retrieving features have been developed by USAL:

- FeatureExtractor for Windows (for image and layout related features)
- JFeatureExtractor (for text related features)

Figure 7 shows the general processing pipeline for preparing features for quality estimation:

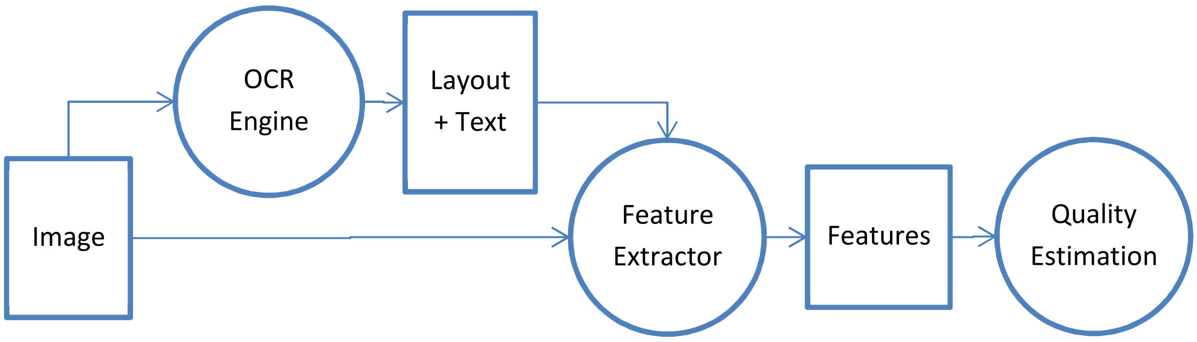


**Figure 7 – Feature extraction and application**

It should be noted that the OCR engine used for feature extraction does not have to be the same as the OCR whose quality is to be estimated. If the target production OCR pipeline is not available (special hardware requirements, license issue) or the result format cannot be used as input for the feature extractor tools, the open source OCR system Tesseract can be used instead. This has been done in the experiments that are detailed in section 3.4 and 3.5.

Table 1 details all features that have been defined and used in the experiments. They range from basic count-based values to results of complex image and text processing operations.

**Table 1 – Image, page layout, and text-related features**

| # | Feature | Description | Type | Range |
|---|---|---|---|---|
| 1 | Colour Mode | Colour mode of the document image:<br>• 1-bit black-and-white (bi-level)<br>• 8-bit greyscale<br>• 24-bit RGB colour | Integer number | 1, 8, 24 |
| 2 | Image DPI | Image resolution in pixels per inch as specified within the document image (only available for TIFF images). Only the horizontal resolution is taken into account. | Fractional number | 72…1200 |
| 3 | Image Tile Count | Feature representing the number of tiles the image can be split into (default tile size 300x300 pixel). The tile size can be changed | Integer number | 1…∞ |

| # | Feature | Description | Type | Range |
|---|---|---|---|---|
| | | using the “-tileSize” command line parameter. Note: Cut off tiles at the edges of the image count as full tiles. | | |
| 4 | Foreground Pixel Density | The number of black pixels within the black-and-white (bi-level) image in relation to the overall image area (0.5 means half of the image is black). | Fractional number | 0…1 |
| 5 | Connected Component Count | Feature representing the average number of connected components within a tile (default 300x300 pixels) of the black-and-white (bi-level) image. | Fractional number | 0…∞ |
| 6 | Image Noise | This feature estimates the noisiness of the document image using a non-local means filter approach. For performance reasons, the feature area is limited to a maximum of 1000x1000 pixels (window at the centre of the image). | Fractional number | 0…1 |
| 7 | Image Brightness | Feature containing the average grey value of the greyscale image. The average grey value is equal to the image brightness. A value of 1 corresponds to a white image whereas a value of 0 corresponds to a fully black image. | Fractional number | 0…1 |
| 8 | Image Contrast | This feature represents the contrast of the greyscale image. It is determined by calculating the standard deviation of the grey value of all pixels. Note: If the original document image is a black-and-white (bi-level) image the contrast is always 1. | Fractional number | 0…1 |
| 9 | Edge Detection | This feature is based on calculating the mean greyscale value of the result image created by the Laplacian operator applied to the greyscale image. The Laplacian operator is used for edge detection. The feature is therefore and indicator for the sharpness of the image (more sharp = more detected edges). | Fractional number | 0…1 |
| 10 | Brightness Unevenness | Feature representing the brightness distribution across the greyscale image. It is calculated as the normalised standard deviation of the brightness of an image tile | Fractional number | 0…1 |

| # | Feature | Description | Type | Range |
|---|---|---|---|---|
| | | (default 300x300 pixel) in relation to the overall image brightness. | | |
| 11 | Layout Region Count | Total number of regions (blocks) in the document page layout.<br>Note: Regions containing child regions are not counted (only top-level regions are regarded). | Integer number | 0…∞ |
| 12 | Text Region Count | Total number of text regions (blocks) in the document page layout.<br>Note: Regions containing child regions are not counted (only top-level regions are regarded). | Integer number | 0…∞ |
| 13 | Region Overlaps | Number of region (block) overlaps (overlaps across different layers are disregarded).<br>Note: Child regions are not taken into account. | Integer number | 0…[number of regions$^2$] |
| 14 | Foreground Outside Regions | The number of black pixels of the black-and-white (bi-level) image that are not within layout regions (blocks).<br>Note: Only top-level regions are taken into account. | Integer number | 0…[image area] |
| 15 | Regions without Foreground | The number of layout regions (blocks) that have less than 1% foreground pixels (black pixels within the black-and-white / bi-level image).<br>Note: Only top-level regions are taken into account. | Integer number | 0…[number of regions] |
| 16 | Missing Region Text | Number of text regions (blocks) that do not have any text content (Unicode). | Integer number | 0…[number of text regions] |
| 17 | Text Line Count Mismatches | Number of text regions (blocks) where the number of text lines within the text content does not match the number of child text line objects.<br>Note: Counts only regions with text content (non-empty) and at least one text line child object. | Integer number | 0…[number of text regions] |
| 18 | OCR Confidence | This feature represents the average text recognition confidence of the OCR engine that was used to analyse the page (e.g. Tesseract). | Fractional number | 0…1 |
| 19 | Word Count | Feature representing the total number of words within the text content. | Integer number | 0…∞ |

| # | Feature | Description | Type | Range |
|---|---|---|---|---|
| 20 | Words with Digits | Feature representing the number of words that contain at least one digit. | Integer number | 0…[word count] |
| 21 | Alphabetic Character Count | Feature representing the count of non-whitespace characters. | Integer number | 0…[text length] |
| 22 | Whitespace Count | Feature representing the number of whitespace characters. | Integer number | 0…[text length] |
| 23 | Digit Count | Feature representing the number of characters that are digits. | Integer number | 0…[text length] |
| 24 | Punctuation Count | Feature representing the number of punctuation characters.<br>Punctuation characters according to POSIX Bracket Expressions:<br>[!"#$%&'()*+,\-./:;<=>?@[\\\]^_`{\|}~] | Integer number | 0…[text length] |
| 25 | Average Word Length | Feature representing the average length in characters of a word within the text content. | Fractional number | 0…[text length] |
| 26 | Words Occurring Once | Feature representing the number of words that occur exactly once in the text in relation to the total number of words. | Fractional number | 0…1 |
| 27 | Word Repetition | Feature representing the number of unique words (excluding repetitions) in relation to the total number of words (including repetitions). | Fractional number | 0…1 |
| 28 | Words in Dictionary | Feature representing the number of words that could be found in the used dictionary in relation to the number of all words. At the moment limited to:<br>• Czech<br>• Dutch<br>• English<br>• German<br>• French<br>• Hungarian<br>• Polish<br>• Russian<br>• Serbian<br>• Swedish<br>• Ukrainian | Fractional number | 0…1 |

### 3.2.3 Combined Features

It can be beneficial to combine two weak features to create one strong one. For example, the features “Words with digits” and “Word count” can be combined to “Words with digits (relative)” by dividing one by the other (expressing the proportion of words in the document containing at least one digit – a high ratio may be valid in a scientific document but not in a book of fiction). Relative values (e.g. ratios) are usually more suitable than absolute values. Furthermore, some complex features can be split into several simpler features (for instance binary – 0/1. Table 2 shows the features that have been used for the experiments.

**Table 2 – Combined and split features**

| # | Feature | Description | Type | Range |
|---|---|---|---|---|
| 29 | Words with Digits (Relative) | Feature representing the number of words that contain at least one digit, divided by the total number of words. | Fractional number | 0…1 |
| 30<br>31<br>32 | Bitonal<br>Greyscale<br>Colour | Splits the feature ‘Colour mode’ into three separate features with 0/1 values. This can be beneficial for some classification methods. | Integer number | 0,1 |

## 3.3 Dataset

The dataset available for experiments consists of the 600 scanned newspaper pages from 12 different institutions that have been produced within the Europeana Newspaper project. Full Ground Truth has been created for page layout (region outlines and types) and text content. Furthermore, all pages have been processed with two OCR workflows:

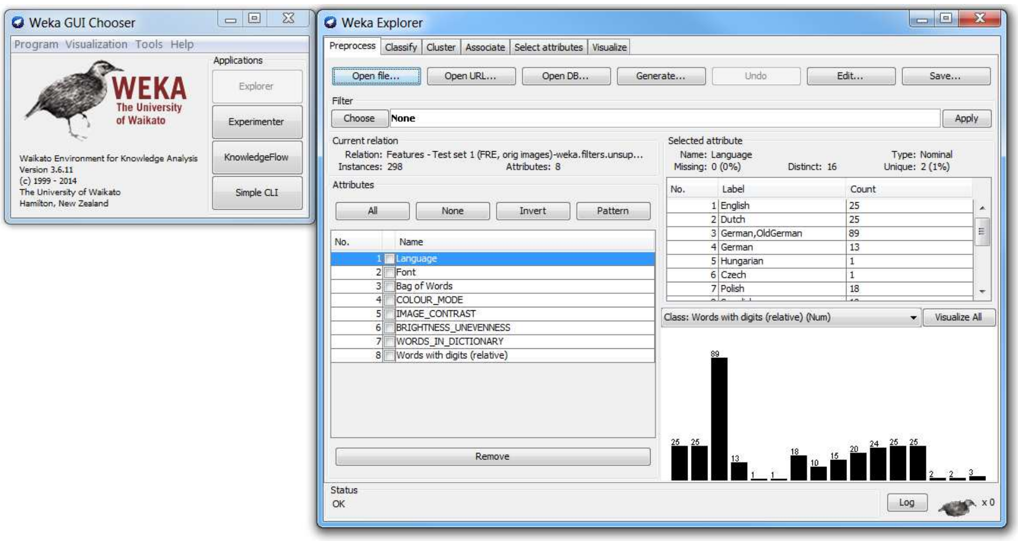
- Binarisation + FineReader Engine
- FineReader Engine only

To create a classifier and test it reliably, the data has to be split into *training* and *test* sets. To this end, 50% of the document pages of each institutional subset have been randomly selected for training. The rest of the pages are used for testing the classifiers. The reason to confine the randomness to each subset (stratification) is to avoid getting a strong bias for one particular subset by chance.

## 3.4 Feature Selection and Classifier Training

A plethora of methods for classification and selecting features have been reported in the literature. The open source tool WEKA (see Figure 8), by the University of Waikato (New Zealand), provides a good selection of standard implementations. It also comes with an excellent user interface for experimentation and even workflow creation.

For the problem at hand, only classifiers that produce a numeric value are of interest, since the target is to predict a quality / success rate. This is also called numeric prediction.


Weka GUI Chooser
Program Visualization Tools Help
WEKA
The University of Waikato
Applications
Explorer
Experimenter
KnowledgeFlow
Simple CLI
Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Open file... Open URL... Open DB... Generate... Undo Edit... Save...
Filter
Choose
Apply
All None Invert Pattern
Remove
Visualize All
Log
x 0

Waikato Environment for Knowledge Analysis
Version 3.6.11
(c) 1999 - 2014
The University of Waikato
Hamilton, New Zealand

Filter: None

Current relation
Relation: Features - Test set 1 (FRE, orig images)-weka.filters.unsup...
Instances: 298 Attributes: 8

Attributes

| No. | | Name |
|---|---|---|
| 1 | ☐ | Language |
| 2 | ☐ | Font |
| 3 | ☐ | Bag of Words |
| 4 | ☐ | COLOUR_MODE |
| 5 | ☐ | IMAGE_CONTRAST |
| 6 | ☐ | BRIGHTNESS_UNEVENNESS |
| 7 | ☐ | WORDS_IN_DICTIONARY |
| 8 | ☐ | Words with digits (relative) |

Selected attribute
Name: Language Type: Nominal
Missing: 0 (0%) Distinct: 16 Unique: 2 (1%)

| No. | Label | Count |
|---|---|---|
| 1 | English | 25 |
| 2 | Dutch | 25 |
| 3 | German,OldGerman | 89 |
| 4 | German | 13 |
| 5 | Hungarian | 1 |
| 6 | Czech | 1 |
| 7 | Polish | 18 |

Class: Words with digits (relative) (Num)

25 25 89 13 1 1 18 10 15 20 24 25 25 2 2 3

Status
OK

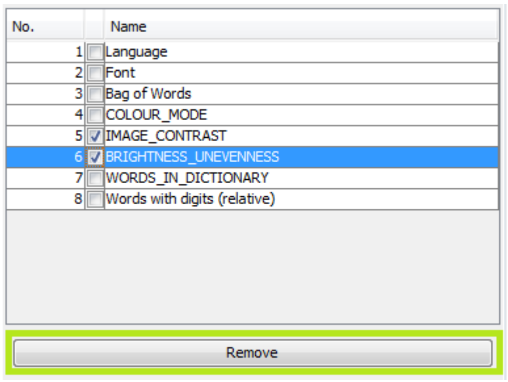
Figure 8 – WEKA Explorer

### 3.4.1 Data Preparation

To be usable in WEKA, the input data has to fulfil following criteria:

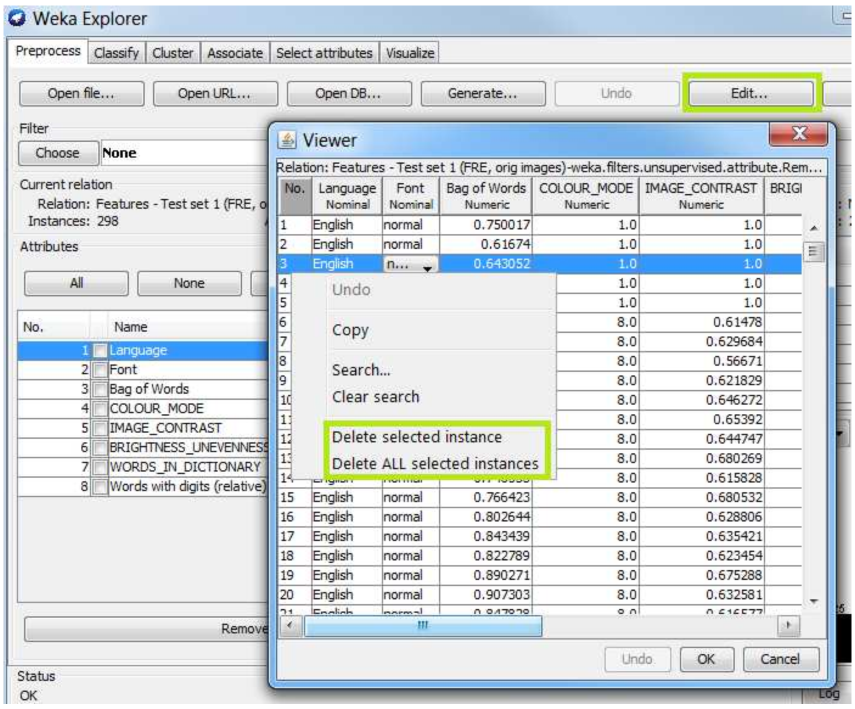
- One table including feature values and target quality values (that are to be estimated by the classifier)
- Clean data (avoid missing values or invalid numbers such as “NaN”)
- Specific WEKA file format (ARFF); can be converted from comma separated values (CSV)

WEKA uses a slightly different vocabulary. Features are called *attributes* and a data table represents the *instances* of the attributes, where one instance equals a row of the data table. The WEKA Explorer allows removing attributes (Figure 9) and instances (Figure 10) from the data. It also supports different data file formats (Figure 11).



| No. | | Name |
|---|---|---|
| 1 | ☐ | Language |
| 2 | ☐ | Font |
| 3 | ☐ | Bag of Words |
| 4 | ☐ | COLOUR_MODE |
| 5 | ✓ | IMAGE_CONTRAST |
| 6 | ✓ | BRIGHTNESS_UNEVENNESS |
| 7 | ☐ | WORDS_IN_DICTIONARY |
| 8 | ☐ | Words with digits (relative) |

**Figure 9 – Remove features / attributes in WEKA Explorer**



Relation: Features - Test set 1 (FRE, orig images)-weka.filters.unsupervised.attribute.Rem...

| No. | Language Nominal | Font Nominal | Bag of Words Numeric | COLOUR_MODE Numeric | IMAGE_CONTRAST Numeric | BRIGI |
|---|---|---|---|---|---|---|
| 1 | English | normal | 0.750017 | 1.0 | 1.0 | |
| 2 | English | normal | 0.61674 | 1.0 | 1.0 | |
| 3 | English | n... | 0.643052 | 1.0 | 1.0 | |
| 4 | | | | 1.0 | 1.0 | |
| 5 | | | | 1.0 | 1.0 | |
| 6 | | | | 8.0 | 0.61478 | |
| 7 | | | | 8.0 | 0.629684 | |
| 8 | | | | 8.0 | 0.56671 | |
| 9 | | | | 8.0 | 0.621829 | |
| 10 | | | | 8.0 | 0.646272 | |
| 11 | | | | 8.0 | 0.65392 | |
| 12 | | | | 8.0 | 0.644747 | |
| 13 | | | | 8.0 | 0.680269 | |
| 14 | | | | 8.0 | 0.615828 | |
| 15 | English | normal | 0.766423 | 8.0 | 0.680532 | |
| 16 | English | normal | 0.802644 | 8.0 | 0.628806 | |
| 17 | English | normal | 0.843439 | 8.0 | 0.635421 | |
| 18 | English | normal | 0.822789 | 8.0 | 0.623454 | |
| 19 | English | normal | 0.890271 | 8.0 | 0.675288 | |
| 20 | English | normal | 0.907303 | 8.0 | 0.632581 | |

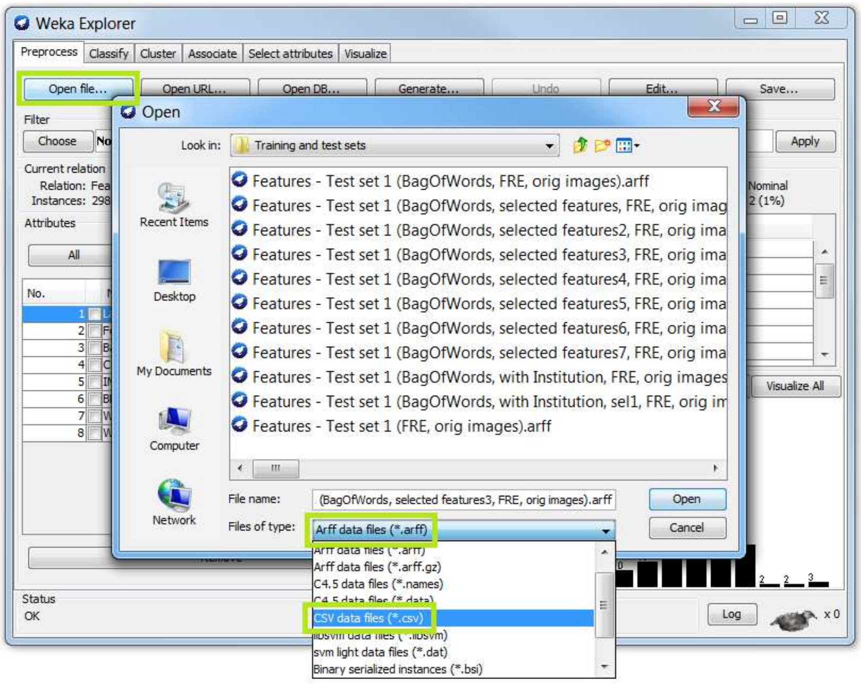
Current relation
Relation: Features - Test set 1 (FRE, o
Instances: 298

Attributes

| No. | Name |
|---|---|
| 1 | Language |
| 2 | Font |
| 3 | Bag of Words |
| 4 | COLOUR_MODE |
| 5 | IMAGE_CONTRAST |
| 6 | BRIGHTNESS_UNEVENNESS |
| 7 | WORDS_IN_DICTIONARY |
| 8 | Words with digits (relative) |

Status
OK

**Figure 10 – Remove instances (data rows) in WEKA Explorer**



Features - Test set 1 (BagOfWords, FRE, orig images).arff
Features - Test set 1 (BagOfWords, selected features, FRE, orig imag
Features - Test set 1 (BagOfWords, selected features2, FRE, orig ima
Features - Test set 1 (BagOfWords, selected features3, FRE, orig ima
Features - Test set 1 (BagOfWords, selected features4, FRE, orig ima
Features - Test set 1 (BagOfWords, selected features5, FRE, orig ima
Features - Test set 1 (BagOfWords, selected features6, FRE, orig ima
Features - Test set 1 (BagOfWords, selected features7, FRE, orig ima
Features - Test set 1 (BagOfWords, with Institution, FRE, orig images
Features - Test set 1 (BagOfWords, with Institution, sel1, FRE, orig im
Features - Test set 1 (FRE, orig images).arff

File name: (BagOfWords, selected features3, FRE, orig images).arff
Files of type: Arff data files (*.arff)

Arff data files (*.arff)
Arff data files (*.arff.gz)
C4.5 data files (*.names)
C4.5 data files (*.data)
CSV data files (*.csv)
libsvm data files (*.libsvm)
svm light data files (*.dat)
Binary serialized instances (*.bsi)

Status
OK

**Figure 11 – WEKA Data file formats**

### 3.4.2 Feature Selection

WEKA offers several heuristics to select the best features for a classifier. After loading training and test data, the classification target has to be selected (e.g. "Bag of Words" OCR success rate).

Extensive experimentation is pointing towards "ClassifierSubsetEval" being the best approach to select features. Thereby a classifier has to be selected beforehand. This method works especially well with the genetic search algorithm. Several feature combinations are tested with the chosen classifier and are then tweaked over many generations (evolutionary optimisation).

Finding good classifiers is a repetitive process of selecting features and classifying using different approaches. For the dataset at hand the classifiers listed in Table 3 were most promising.

**Table 3 – Examples of classifiers implemented in WEKA**

| Classifier (WEKA name) | Description |
|---|---|
| GaussianProcesses | Implements Gaussian Processes for regression without hyperparameter-tuning. For more information see:<br>David J.C. Mackay (1998). Introduction to Gaussian Processes. Dept. of Physics, Cambridge University, UK. |
| LinearRegression | Class for using linear regression for prediction. Uses the Akaike criterion for model selection, and is able to deal with weighted instances. |
| SMOReg | Implements the support vector machine for regression. The parameters can be learned using various algorithms. The algorithm is selected by setting the RegOptimizer. The most popular algorithm (RegSMOImproved) is due to Shevade, Keerthi et al and this is the default RegOptimizer. For more information see:<br>S.K. Shevade, S.S. Keerthi, C. Bhattacharyya, K.R.K. Murthy: Improvements to the SMO Algorithm for SVM Regression. In: IEEE Transactions on Neural Networks, 1999. |
| IBk | K-nearest neighbours classifier. For more information, see:<br>D. Aha, D. Kibler (1991). Instance-based learning algorithms. Machine Learning. 6:37-66. |

Table 4 shows a few examples of selected features to predict the "Bag of Words" quality measure for text recognition.

**Table 4 – Examples of different feature selections**

| Target classifier (ClassifierSubsetEval with genetic search) | Selected Features |
|---|---|
| Gaussian Processes | Language, Image DPI, Image tile count, Text region count, OCR confidence, Alphabetic character count, Words occurring once, Words in dictionary, Greyscale |
| SMOReg (Support vector machine) | Language, Image DPI, Image brightness, Edge detection, Layout region count, Text region count, OCR confidence, Average word length, Words occurring once, Word repetition, Words in dictionary, Words with digits (relative), Bitonal |
| Multi-layer perceptron | Font, Colour mode, Image DPI, Foreground pixel density, Connected component count, Brightness unevenness, Layout region count, Missing region text, OCR confidence, Words with digits, Whitespace count, Digit count, Word repetition, Words in dictionary |
| IBk | Language, Image DPI, Image tile count, Layout region count, Text region count, OCR confidence, Word count, Alphabetic character count, Punctuation count, Words occurring once, Word repetition, Words in dictionary, Words with digits (relative) |

Based on several feature selection iterations, the usefulness of specific features could be estimated by counting how often each individual feature was selected. Table 5 shows the features that were selected most are (in order of relevance). The least relevant features (in this use scenario) are listed in Table 6.

**Table 5 – Most used features (features occurring in both columns are highlighted)**

| Best features for "Bag of Words" prediction | Best features for layout analysis quality prediction |
|---|---|
| **Words in dictionary** | **Foreground pixel density** |
| Words occurring once | Edge detection |
| **OCR confidence** | Whitespace count |
| Language | **Word repetition** |
| Image DPI | Image contrast |
| Layout region count | **OCR confidence** |
| **Words with digits (relative)** | Alphabetic character count |
| **Text region count** | **Words with digits (relative)** |
| **Word repetition** | Words occurring once |
| Font | **Text region count** |

| | |
|---|---|
| **Foreground pixel density** | **Punctuation count** |
| **Punctuation count** | Image noise |
| | Layout region count |
| | Word count |
| | **Words in dictionary** |
| | Greyscale |

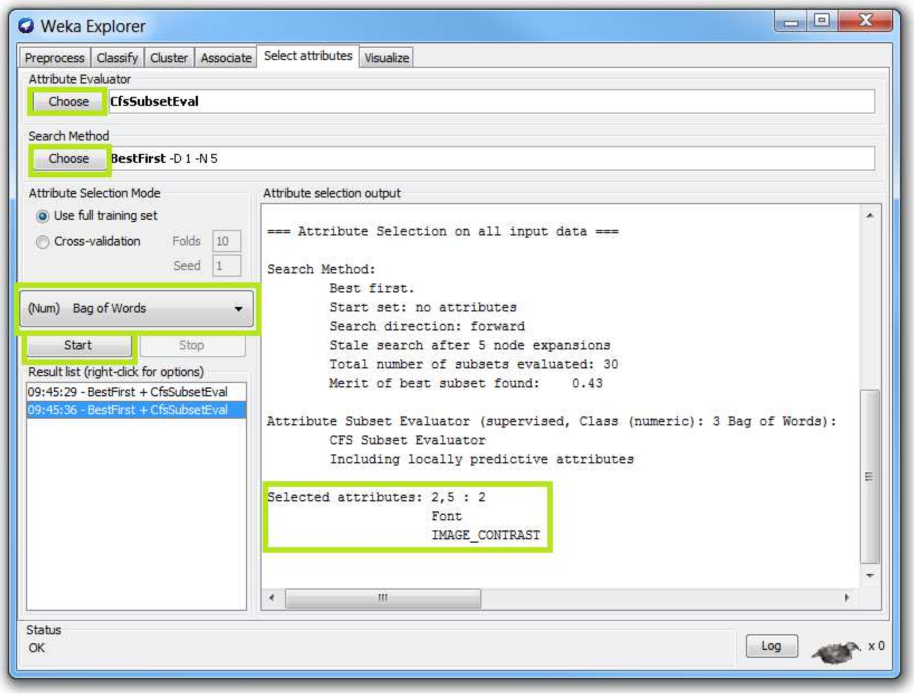
Table 6 – Least used features

| Least relevant features for "Bag of Words" prediction | Least relevant features for layout analysis quality prediction |
|---|---|
| Regions without foreground | Average word length |
| Colour | |

#### *3.4.2.1 Selecting attributes (features) in WEKA*

The WEKA Explorer tool provides a flexible user interface to easily test several feature selection approaches (see Figure 12). These are the main steps to follow:

1. Choose an evaluator and adjust its settings (by clicking on the text field with the name)
2. Choose a search method and adjust its settings (by clicking on the text field with the name)
3. Important: Select the target attribute (the quality measure that is to be predicted later on)
4. Start the selection process
5. Find the selected features in the output text area
6. Save copies of training and test set files with only the selected features



Attribute Evaluator: CfsSubsetEval

Search Method: BestFirst -D 1 -N 5

Attribute Selection Mode: (●) Use full training set ○ Cross-validation Folds 10 Seed 1

(Num) Bag of Words

Result list (right-click for options)
09:45:29 - BestFirst + CfsSubsetEval
09:45:36 - BestFirst + CfsSubsetEval

Attribute selection output

```
=== Attribute Selection on all input data ===

Search Method:
        Best first.
        Start set: no attributes
        Search direction: forward
        Stale search after 5 node expansions
        Total number of subsets evaluated: 30
        Merit of best subset found:    0.43

Attribute Subset Evaluator (supervised, Class (numeric): 3 Bag of Words):
        CFS Subset Evaluator
        Including locally predictive attributes

Selected attributes: 2,5 : 2
                     Font
                     IMAGE_CONTRAST
```

Status
OK

**Figure 12 – Feature selection in WEKA Explorer**

### 3.4.3 Classifier Training

Classifiers are trained purely on the training set and afterwards evaluated using the test set. To state the success of a classifier we use the mean error (absolute difference between predicted and actual value). Since we want to predict OCR and layout evaluation quality, the mean error can be expressed as a percentage. For instance, a mean error of 12% means that, in average, the predicted result differs 12% from the actual result (0% would be the optimum).

As a baseline for comparison, we also calculate the mean error for a naïve prediction. To obtain it, a fixed prediction value is calculated based on the training set simply as the average of the target (segmentation, OCR etc.) actual quality values. That fixed value is then used as “prediction” for all instances in the test set. The mean error of this approach can be seen as the best result a quality estimation method with fixed prediction value can achieve. A trained classifier should therefore outperform the naïve approach, to be considered successful.

WEKA provides several classifiers for numeric prediction, of which a Support Vector Machine for Regression (SMOReg) and Gaussian Processes delivered the best results. Table 7 states some results for different prediction targets, data subsets, feature selections, and classifiers. For these experiments *Tesseract OCR* results have been used for feature extraction. The prediction targets however, are the quality of layout analysis and text recognition results of *ABBYY FineReader*.

**Table 7 – Results of selected experiments**

| Prediction target | Dataset (#pages in training and test set) | Baseline (mean error of naïve prediction) | Features | Classifier | Mean error of prediction (test set) |
|---|---|---|---|---|---|
| Bag of Words OCR Success Rate | Full (300+300) | 14.2% | Language, Image DPI, Image tile count, Text region count, OCR confidence, Alphabetic character count, Words occurring once, Words in dictionary, Greyscale | Gaussian Processes | 6.2% |
| | | | Language, Image DPI, Image brightness, Edge detection, Layout region count, Text region count, OCR confidence, Average word length, Words occurring once, Word repetition, Words in dictionary, Words with digits (relative), Bitonal | Support vector machine | 6.1% |
| | | | All features | Gaussian Processes | 7.1% |
| | English documents (25+25) | 8.1% | Image noise, Image contrast, Words in dictionary | Support vector machine | 2.67% |
| | Dutch documents (25+25) | 11.2% | Image contrast, Edge detection, Layout region count, Text region count, Punctuation count, Average word length, Words occurring once, Word repetition, Words in dictionary, Words with digits (relative) | IBk | 2.73% |
| Layout Analysis Success Rate (Scenario: Keyword search) | Full (300+300) | 14.5% | Colour mode, Foreground pixel density, Image noise, Edge detection, Image tile count, Text region count, Region overlaps, Foreground outside regions, OCR confidence, Word count, Words with digits, Alphabetic character count, Whitespace count, Punctuation | Support vector machine | 11.27% |

| | | | | |
|---|---|---|---|---|
| | | count, Words occurring once, Bitonal | | |
| English documents (25+25) | 7.8% | Image DPI, Foreground pixel density, Image noise, Image brightness, Image contrast, Image tile count, Regions without foreground, OCR confidence, Words occurring once, Word repetition | Gaussian Processes | 4.42% |

To get a better understanding of these figures, a colour coded list of results for individual pages has been produced for the subset of Dutch documents (training set); see Table 8. Green cells means predicted and actual values are almost equal (close enough). Red cells highlight instances with large mean error. Since it can be of importance towards what direction the prediction is wrong, overprediction errors have been marked red and underprediction blue.

**Table 8 – Results per document page for Bag-of-words experiment on Dutch documents**

| Instance | Actual quality | Predicted quality | Error | Absolute error |
|---|---|---|---|---|
| 1 | 45.3% | 46.4% | 1.1% | 1.1% |
| 2 | 63.9% | 69.2% | 5.3% | 5.3% |
| 3 | 90.3% | 91.2% | 0.9% | 0.9% |
| 4 | 87.8% | 88.8% | 1.0% | 1.0% |
| 5 | 80.0% | 79.1% | -1.0% | 1.0% |
| 6 | 95.9% | 94.7% | -1.2% | 1.2% |
| 7 | 54.5% | 46.4% | -8.1% | 8.1% |
| 8 | 97.4% | 94.7% | -2.7% | 2.7% |
| 9 | 90.9% | 93.3% | 2.5% | 2.5% |
| 10 | 92.6% | 93.3% | 0.7% | 0.7% |
| 11 | 63.8% | 69.2% | 5.4% | 5.4% |
| 12 | 89.8% | 96.2% | 6.4% | 6.4% |
| 13 | 97.9% | 97.6% | -0.3% | 0.3% |
| 14 | 91.8% | 96.8% | 4.9% | 4.9% |
| 15 | 93.4% | 94.0% | 0.6% | 0.6% |
| 16 | 49.5% | 46.4% | -3.1% | 3.1% |
| 17 | 89.7% | 90.2% | 0.5% | 0.5% |
| 18 | 95.5% | 96.2% | 0.7% | 0.7% |
| 19 | 86.3% | 90.2% | 3.9% | 3.9% |
| 20 | 86.4% | 79.1% | -7.3% | 7.3% |
| 21 | 89.8% | 90.2% | 0.4% | 0.4% |
| 22 | 92.4% | 92.0% | -0.3% | 0.3% |

| | | | | |
|---|---|---|---|---|
| 23 | 95.3% | 94.0% | -1.3% | 1.3% |
| 24 | 84.7% | 88.5% | 3.8% | 3.8% |
| 25 | 74.0% | 79.1% | 5.1% | 5.1% |

From the results it can be observed that the quality estimation is more precise for smaller datasets with more similar documents (see results for English and Dutch subsets). The prediction performance for the full dataset is with an average error of 6.1% still acceptable (for Bag of Words). Figure 13 shows the distribution of the errors values for all document pages. The negative values on the left mean underprediction and the positive values overprediction.

Predicting the layout analysis performance seems to be a much harder problem than predicting text recognition results. The best average error that could be achieved for the full dataset is 11.3%. Future research for better features can improve the quality estimation considerably.

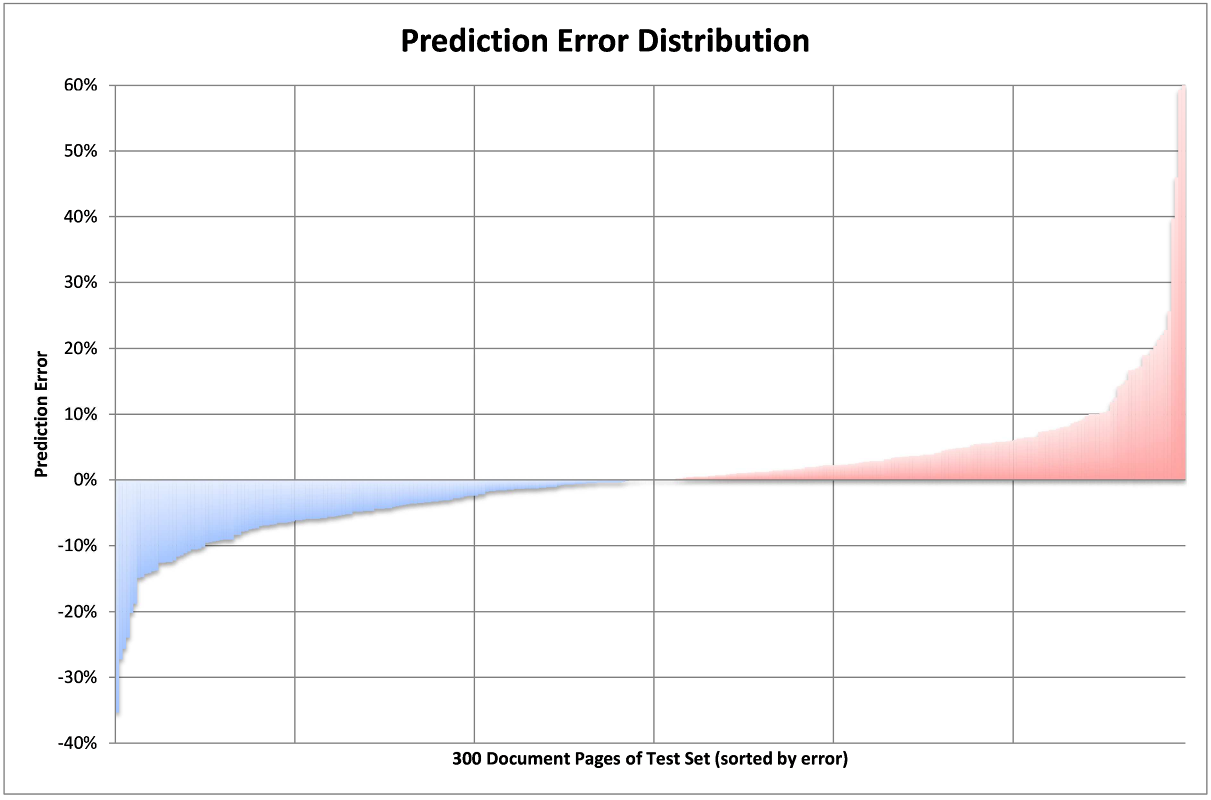


**Figure 13 – Error distribution for the 300 documents of the test set (BagOfWords prediction using support vector machine).**

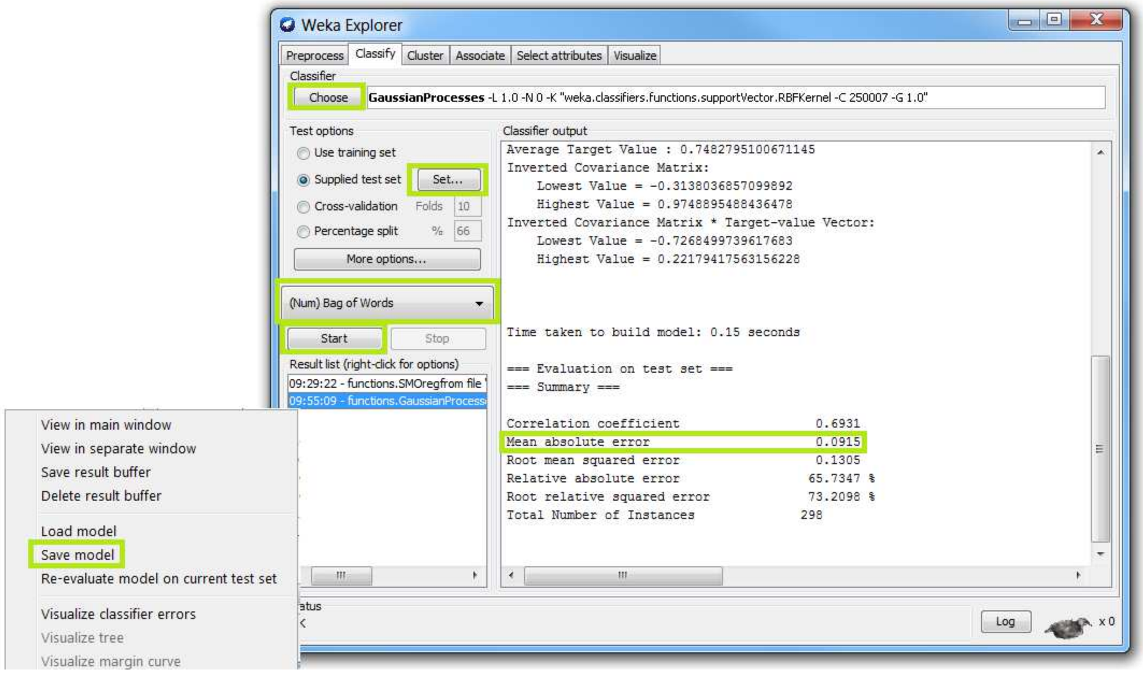
Additional experiments have been carried out using ABBYY FineReader Engine for both feature extraction and as the prediction target. This use scenario is limited by the factor that the feature extraction tools require PAGE XML as input. The OCR results therefore have to be either exported directly in this format (e.g. by using USAL's FineReader Integration tool)

or existing results (e.g. in ALTO format) have to be converted, which might involve loss of useful information.
While it could be expected that using the same OCR engine for feature extraction and as prediction target might result in a much more precise prediction, the outcome of the experiments only show a minimal improvement compared to using Tesseract. The “Bag of Words” prediction error, for instance, is reduced by only 0.03% (from 6.12% using Tesseract to 6.09% using FineReader).

#### *3.4.3.1 Training a classifier in WEKA*

Once features have been selected and data files (training and test set) with those features have been created, different classifiers can be trained using the WEKA Explorer (Figure 14). Following steps are required:

1. Load the training set with the selected features
2. Choose a classifier and adjust its settings (by clicking of the text field with the name)
3. Select the corresponding test set with the same selected features
4. Select the target attribute (the quality measure that is to be predicted)
5. Run the training
6. Check the result in the output text field (e.g. ‘Mean absolute error’)
7. Save the classifier model to be able to use it later without training



GaussianProcesses -L 1.0 -N 0 -K "weka.classifiers.functions.supportVector.RBFKernel -C 250007 -G 1.0"

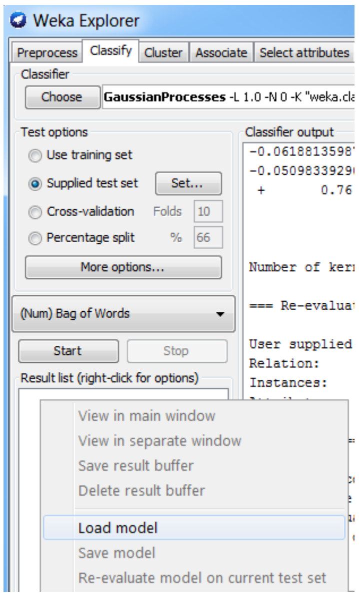
```
Average Target Value : 0.7482795100671145
Inverted Covariance Matrix:
    Lowest Value = -0.3138036857099892
    Highest Value = 0.9748895488436478
Inverted Covariance Matrix * Target-value Vector:
    Lowest Value = -0.7268499739617683
    Highest Value = 0.22179417563156228



Time taken to build model: 0.15 seconds

=== Evaluation on test set ===
=== Summary ===

Correlation coefficient                  0.6931
Mean absolute error                      0.0915
Root mean squared error                  0.1305
Relative absolute error                 65.7347 %
Root relative squared error             73.2098 %
Total Number of Instances              298
```

**Figure 14 – Classifier training in WEKA Explorer**

It is also possible to load a previously trained classifier and output the predictions for each data instance:

1. Load the dataset with the extracted features (in ‘Preprocess’ tab)
2. Load the classifier model (Figure 15)

3. Select the dataset again as the ‘Supplied test set’
4. Enable the output of the predictions (Figure 16)
5. Select the target attribute (the quality measure that is to be predicted)
6. Run the classification (Figure 17)
7. See the results in the output text field (Figure 18)
8. Optional: Save the result buffer and import into a spread sheet (e.g. Microsoft Excel) for further analysis

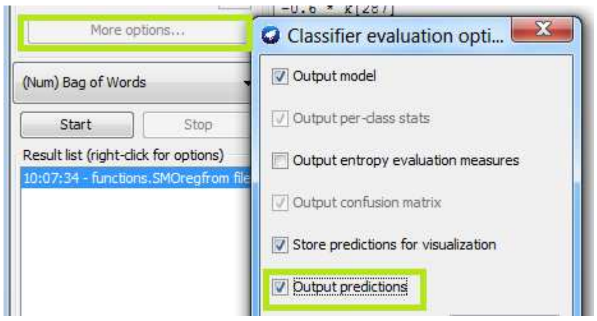


GaussianProcesses -L 1.0 -N 0 -K "weka.cla

Folds 10 | % 66 | (Num) Bag of Words

```
-0.0618813598
-0.0509833929
 +        0.76


Number of ker

=== Re-evalua

User supplied
Relation:
Instances:
```

Figure 15 – Loading and using a classifier in WEKA Explorer


More options...
Start
Stop
Result list (right-click for options)
Classifier evaluation opti...

(Num) Bag of Words

10:07:34 - functions.SMOregfrom file

[✓] Output model
[✓] Output per-class stats
[ ] Output entropy evaluation measures
[✓] Output confusion matrix
[✓] Store predictions for visualization
[✓] Output predictions

Figure 16 – Output predictions in WEKA Explorer

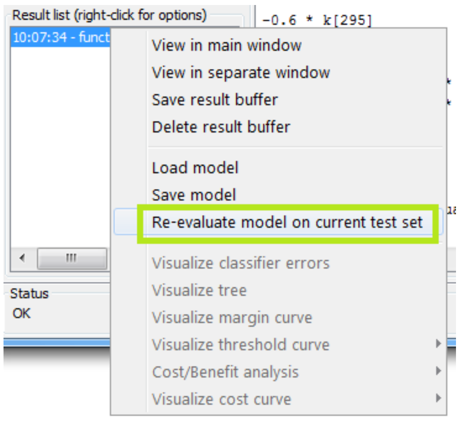
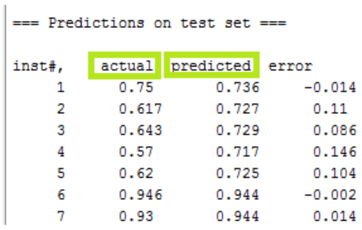


**Figure 17 – Classify test set instances in WEKA Explorer**



```
=== Predictions on test set ===

inst#,    actual   predicted   error
     1     0.75      0.736     -0.014
     2     0.617     0.727      0.11
     3     0.643     0.729      0.086
     4     0.57      0.717      0.146
     5     0.62      0.725      0.104
     6     0.946     0.944     -0.002
     7     0.93      0.944      0.014
```

**Figure 18 – Prediction results in WEKA Explorer**

## 3.5 Predicting Performance

A new Quality Estimation command line tool has been developed by USAL that can predict the OCR result quality of a single document image, using a WEKA classifier. Alternatively, a built-in workflow creation tool within WEKA can be used to run a prediction using a previously trained classifier.

### 3.5.1 Quality Estimation Tool

The PRImA Quality Estimation command line tool has been developed as part of the ENP project to predict the quality of page analysis methods (including OCR) using a previously trained classifier. The tool uses the open source WEKA library for classification (numeric prediction).

A typical use scenario comprises two pipelines – one for training and one for the actual quality estimation (see Figure 19).

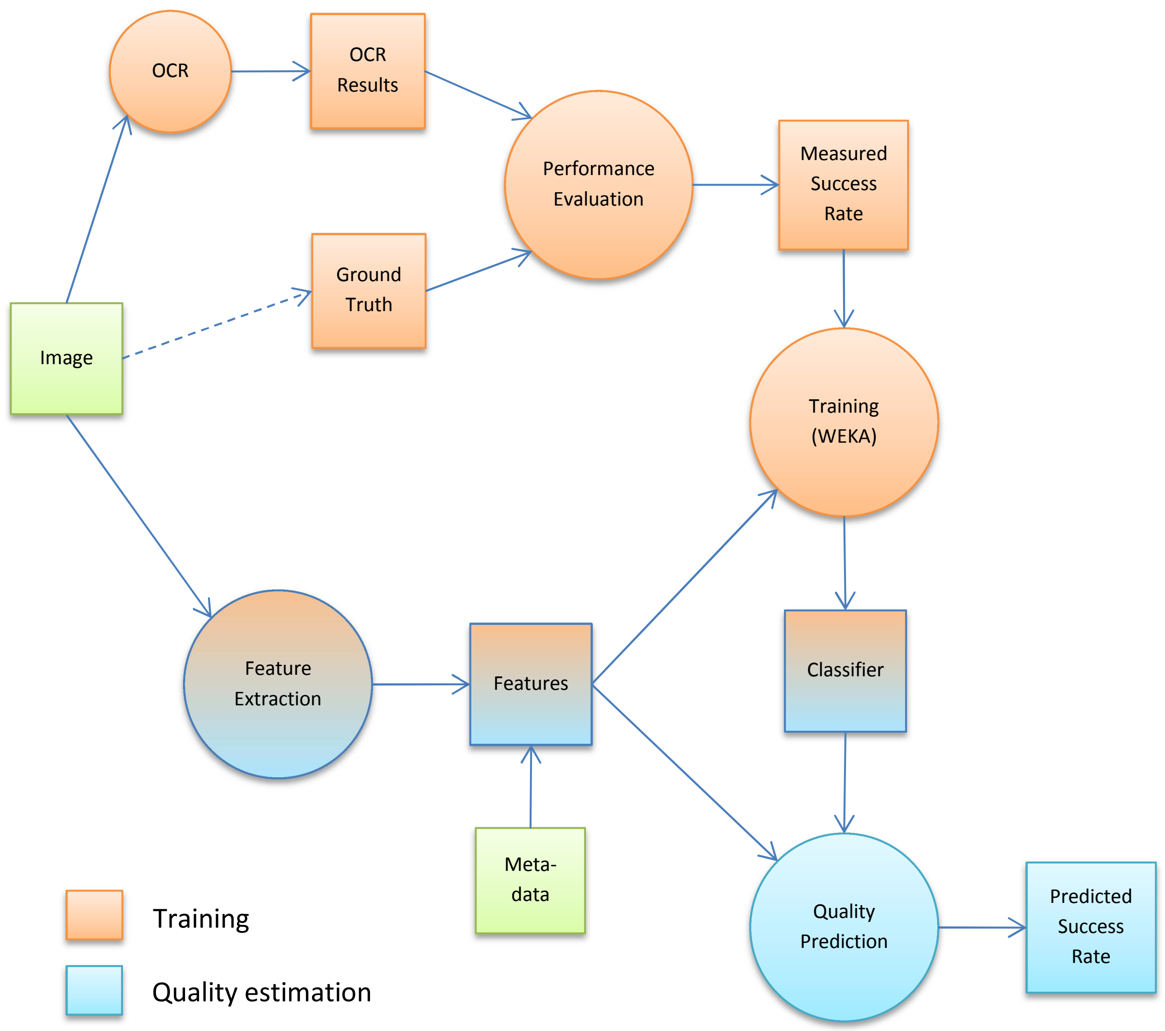


**Figure** 19 **– Training and prediction workflow**

The tool has been designed to determine from the given classifier model which features are required for the quality prediction. It then analyses the provided input sources (CSV with pre-computed features, direct feature input, OCR result, and document image) and extracts the missing features if necessary, by running an integrated OCR engine and/or using feature extraction methods. To this end, the Quality Prediction tool is linked with several other tools developed by USAL:

- TesseractToPAGE tool (a wrapper for Tesseract OCR engine)
- Page Converter (to apply optional text filter rules)
- Text Exporter (to serialise the text content of OCR results in PAGE format)
- Feature extractors

The estimated quality value is output directly to the command line and can be added to a text file, for instance. In addition, a file containing a table with all features and the quality value can be produced optionally (WEKA file format). Figure 20 provides a schematic overview of the functioning of the tool.

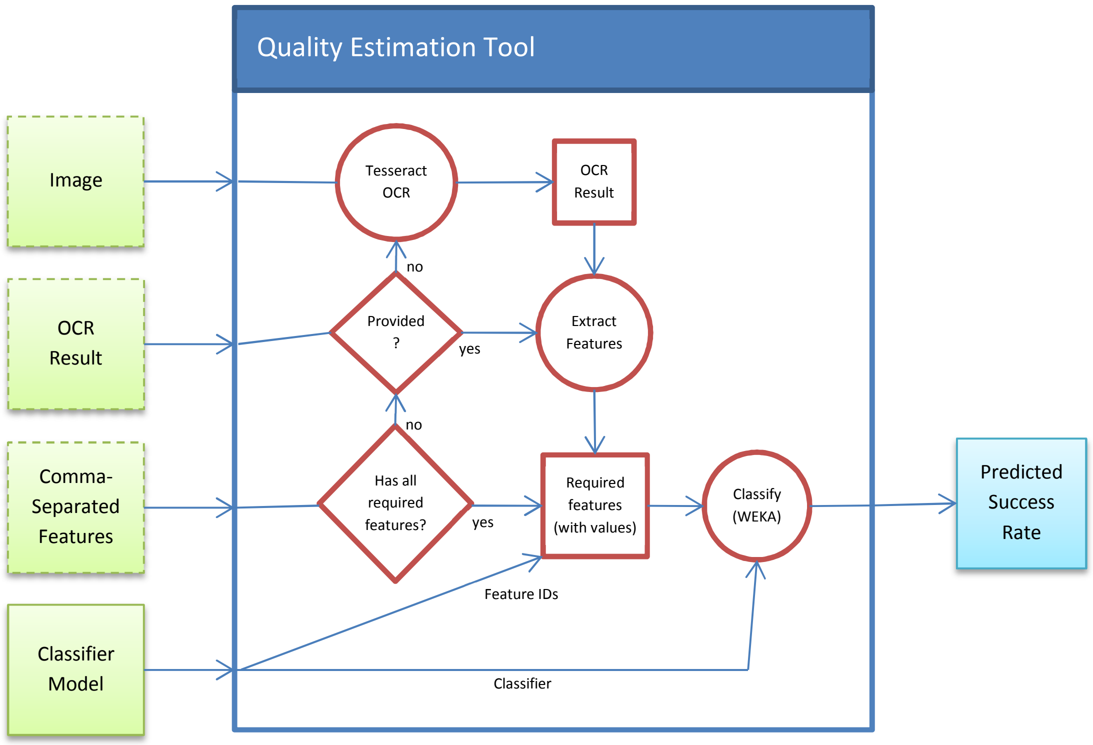
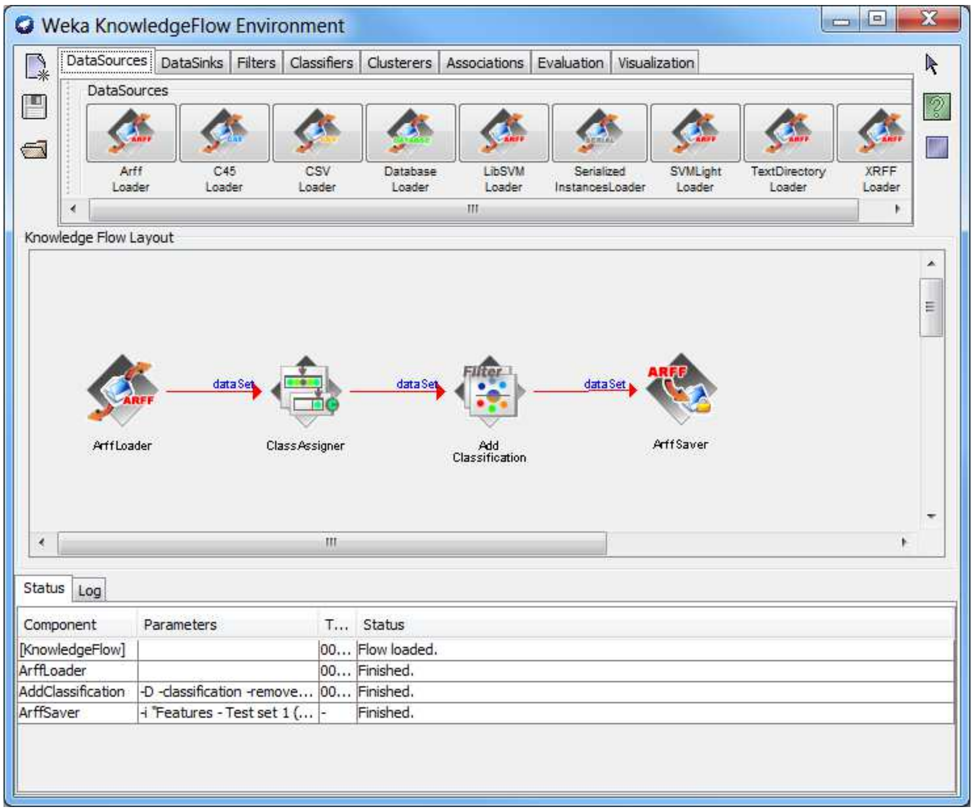


**Figure 20 – Diagram of PRImA Quality Estimation tool**

### 3.5.2 WEKA KnowledgeFlow for Performance Prediction

The WEKA tool suite contains KnowledgeFlow (see Figure 21), which can be used to create automated workflows for all tasks that can be done manually in WEKA Explorer. Following steps can be used to create a workflow:

1. Open the KnowledgeFlow tool
2. Add an ‘ArffLoader’ processing node and select the ARFF file with the extracted features of your dataset (also add an empty attribute for the prediction result)
3. Add a ‘ClassAssigner’ node and select the empty attribute
4. Add an ‘Add Classification’ node, select the saved classifier model, and enable ‘outputClassification’ and ‘removeOldClass’
5. Add an ‘ArffSaver’ node and select the output destination
6. Connect all nodes (right click – dataset)
7. Start the workflow (right click on ArffLoader node – Start loading)

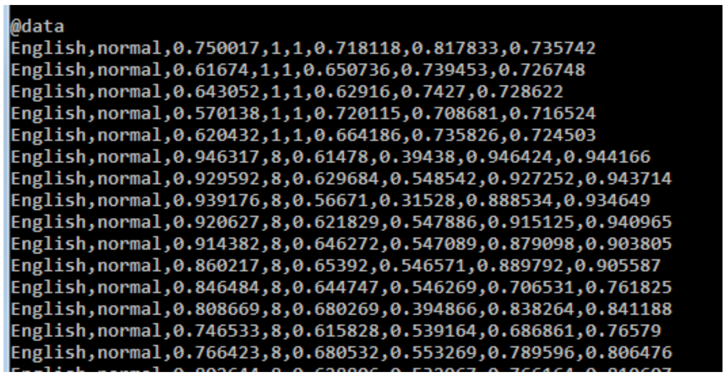


| Component | Parameters | T... | Status |
|---|---|---|---|
| [KnowledgeFlow] | | 00... | Flow loaded. |
| ArffLoader | | 00... | Finished. |
| AddClassification | -D -classification -remove... | 00... | Finished. |
| ArffSaver | -i "Features - Test set 1 (... | - | Finished. |

Figure 21 – WEKA KnowledgeFlow tool

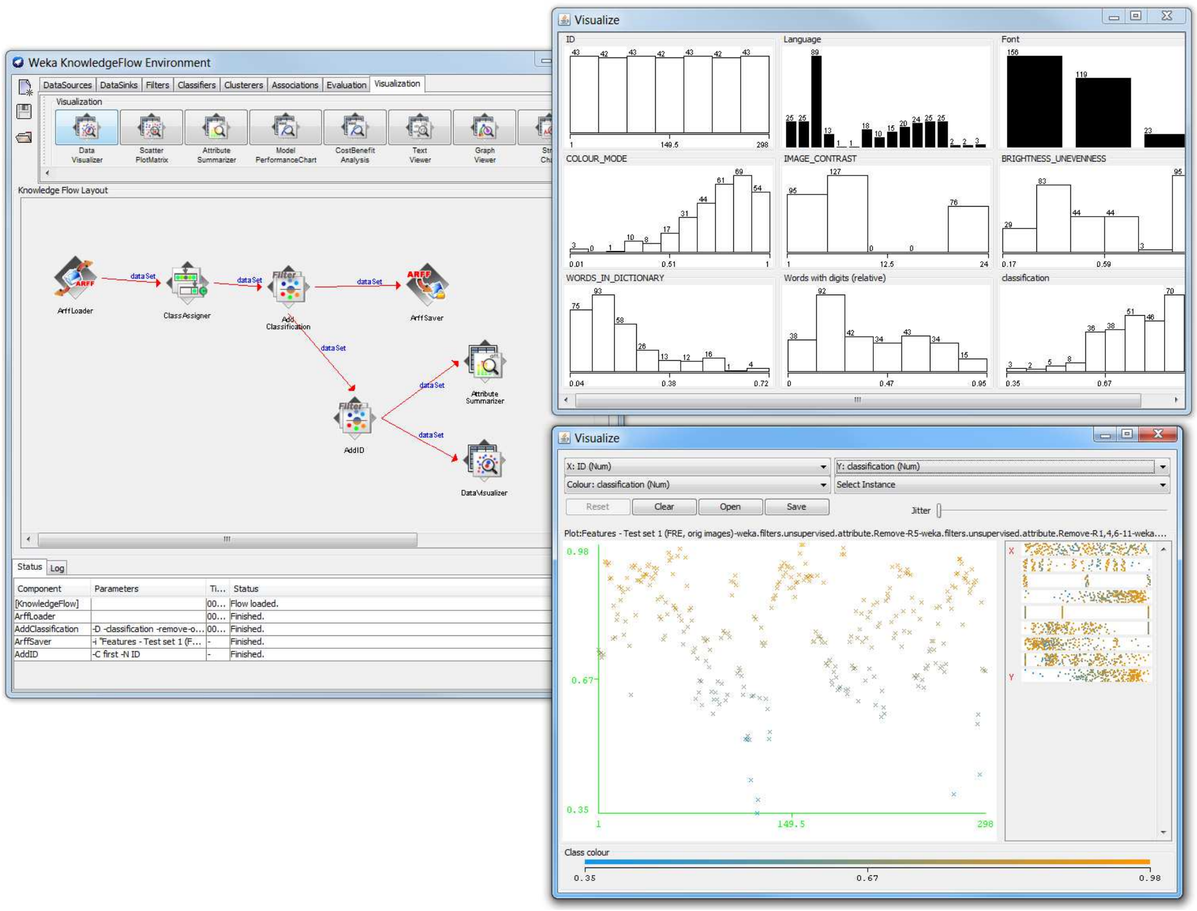
Note: If the ArffSaver node doesn't output a file, run WEKA with a command line console and check the output. If the saving fails, the result file is output to the command line window (Figure 22).

Furthermore, it is possible to extend the workflow with visualizers, see Figure 23 for an example.



```
@data
English,normal,0.750017,1,1,0.718118,0.817833,0.735742
English,normal,0.61674,1,1,0.650736,0.739453,0.726748
English,normal,0.643052,1,1,0.62916,0.7427,0.728622
English,normal,0.570138,1,1,0.720115,0.708681,0.716524
English,normal,0.620432,1,1,0.664186,0.735826,0.724503
English,normal,0.946317,8,0.61478,0.39438,0.946424,0.944166
English,normal,0.929592,8,0.629684,0.548542,0.927252,0.943714
English,normal,0.939176,8,0.56671,0.31528,0.888534,0.934649
English,normal,0.920627,8,0.621829,0.547886,0.915125,0.940965
English,normal,0.914382,8,0.646272,0.547089,0.879098,0.903805
English,normal,0.860217,8,0.65392,0.546571,0.889792,0.905587
English,normal,0.846484,8,0.644747,0.546269,0.706531,0.761825
English,normal,0.808669,8,0.680269,0.394866,0.838264,0.841188
English,normal,0.746533,8,0.615828,0.539164,0.686861,0.76579
English,normal,0.766423,8,0.680532,0.553269,0.789596,0.806476
```

Figure 22 – Output of quality estimation workflow using WEKA KnowledgeFlow



| Component | Parameters | Ti... | Status |
|---|---|---|---|
| [KnowledgeFlow] | | 00... | Flow loaded. |
| ArffLoader | | 00... | Finished. |
| AddClassification | -D -classification -remove-o... | 00... | Finished. |
| ArffSaver | -i "Features - Test set 1 (F... | - | Finished. |
| AddID | -C first -N ID | - | Finished. |

X: ID (Num)
Y: classification (Num)
Colour: classification (Num)

Plot:Features - Test set 1 (FRE, orig images)-weka.filters.unsupervised.attribute.Remove-R5-weka.filters.unsupervised.attribute.Remove-R1,4,6-11-weka....

**Figure 23 – Visualisation of prediction results using WEKA KnowledgeFlow tool**

## 3.6 Discussion

A comprehensive workflow for quality estimation has been described, based on open source software and tools newly developed by USAL. Predicting the outcome of digitisation pipelines, based on machine learning techniques, can complement pilot projects or can be used for pre-selection of data (triage).

Experiments carried out on the Europeana Newspapers dataset showed both the potential as well as the limitations of quality estimation in general. Given a large enough training set and a test set without too much variation, the quality prediction can be quite accurate. On the other hand, small training sets and strong variation will not produce meaningful results. Examples are the subset of English documents, which lead to a reasonably good *prediction accuracy* of 97.3% (for Bag of Words), and the full dataset of 600 pages which delivered only 93.9% prediction accuracy.

A large selection of features (based on image, page layout, and text content) has been proposed and feature extraction tools have been implemented. Surprising was, as shown by the experiments, that even the most basic features (such as region count) can be very useful for the prediction. More features can be added in the future; the software tools have been designed to accommodate this. In addition, current features could be enhanced, for instance by adding dictionaries for more languages ("words in dictionary" feature) or extending existing dictionaries.

The WEKA tool is excellent for testing different approaches and setting up experiments. It is suitable for non-experts, given appropriate guidance. Nevertheless, using it for feature selection and classifier training requires a significant amount of manual labour. Future work could integrate those tasks into the Quality Estimation tool, thereby concentrating on the most promising methods.

# 4 Concluding Remarks

This report presented two practical aspects of performance evaluation that can be used to obtain actionable information in order to make informed decisions on planning and commissioning digitisation projects. First, a workflow for the evaluation of pilot projects is presented. It combines earlier findings mainly from Tasks 3.1 (use scenarios), 3.2 (evaluation datasets), 3.3 (evaluation tools), and 3.5 (performance evaluation) into a coherent set of steps to realise a pilot project. The resources are available (software tools and data repositories) to achieve this are described and a topical example from the project is presented.

Second, an approach (complete with software tools) for estimating the performance of digitisation pipelines is introduced. As experimental evidence shows, the quality of digitisation results can be predicted reasonably accurately, using concepts from machine learning, based on only the scanned pages (with minimal training). Quality estimation can complement pilot projects and can be used for purposes of triage (selection of documents to be included in digitisation projects).