Historical Typewritten Document Recognition Using Minimal User Interaction

George Retsinas[1,2], Basilis Gatos[1], Apostolos Antonacopoulos[3], Georgios Louloudis[1] and Nikolaos Stamatopoulos[1]

[1] Computational Intelligence Laboratory, Institute of Informatics and Telecommunications
National Center for Scientific Research "Demokritos"
GR-15310 Athens, Greece
{georgeretsi,bgat,louloud,nstam}@iit.demokritos.gr

[2] School of Electrical and Computer Engineering
National Technical University of Athens
GR-15773 Athens, Greece

[3] Pattern Recognition and Image Analysis (PRImA) Research Lab
School of Computing, Science and Engineering,University of Salford
M5 4WT Greater Manchester, United Kingdom
A.Antonacopoulos@primaresearch.org

ABSTRACT

Recognition of low-quality historical typewritten documents can still be considered as a challenging and difficult task due to several issues i.e. the existence of faint and degraded characters, stains, tears, punch holes etc. In this paper, we exploit the unique characteristics of historical typewritten documents in order to propose an efficient recognition methodology that requires minimum user interaction. It is based on a pre-processing stage in order to enhance the quality and extract connected components, on a semi-supervised clustering for detecting the most representative character samples and on a segmentation-free recognition stage based on a template matching and cross-correlation technique. Experimental results prove that even with minimum user interaction, the proposed method can lead to promising accuracy results.

CCS Concepts

•**Applied computing** → **Optical character recognition;** •**Theory of computation** → *Unsupervised learning and clustering;*

Keywords

Historical Typewritten Document Analysis and Recognition; Character Clustering; Semi-supervised Clustering

HIP '15 August 22, 2015, Gammarth, Tunisia

DOI: 10.1145/1235

1. INTRODUCTION

Historical typewritten documents can be considered an important source of novel information for scholars in order to study recent history. Large collections of typewritten documents still remain unexploited since existing recognition techniques cannot be successfully applied to the cases of low-quality historical typewritten documents that suffer from several problems such as the existence of faint and degraded characters, broken or merged characters, stains, tears and punch holes, quality problems due to ageing. Moreover, a great part of these collections may survive only as a carbon copy of the original and this poses additional challenges concerning image quality.

Existing approaches related to the analysis and recognition of historical typewritten documents mainly focus on the enhancement of the degraded typewritten characters before using a conventional OCR engine [1–4]. As it is reported in [3, 4], the recognition accuracy can be increased from 77.2% to 91.3% [3] or from 54.5% to 78.2% [4] when a suitable enhancement of text areas is applied. A new framework for recognizing particularly challenging collections of historical documents, such as typewritten documents of World War II is proposed in [5]. A recognition approach is proposed which can be trained on specific document collections (e.g. produced by similar typewriters) with minimal user interaction. It is based on the combination of collection-independent domain knowledge (such as typography conventions) with human feedback in an iterative manner to gradually refine the system's understanding of the unique characteristics of the specific document collections.

This paper is based on the basic principle of [5] and mainly focuses on (i) having minimum but effective user-interaction, (ii) introducing a robust clustering scheme !! and (iii) involving a segmentation-free recognition module in order to overcome segmentation problems. User-interaction is a common practice in many OCR applications in order to correct the result and re-train the corresponding classifiers. In [6], a "carpet session" is involved in order to sort all detected

characters in a way that similar characters appear close to each other. The user has to identify all errors and thus automatically approve all other characters as valid. To eliminate this tedious process, we propose to involve a sub-clustering procedure in order to have the user checking only a limited number of representative characters (e.g. 4) per character class. The exact re-arrangement of the clusters/classes is done via a semi-supervised clustering scheme which uses the provided information from the user as a rough labeling. The workflow of the proposed methodology is presented in Figure 1 and its components are described in detail in the upcoming sections.

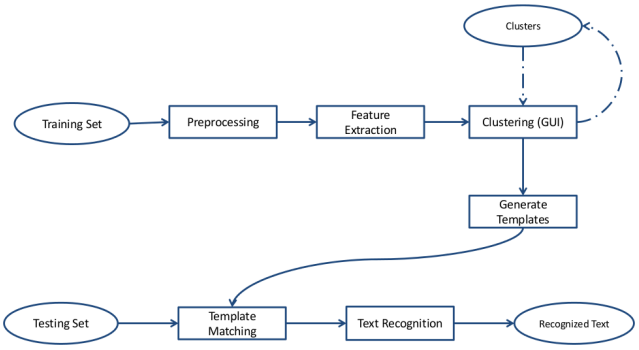


Figure 1: System Overview

The remainder of the paper is organized as follows. In Section 2, the preprocessing steps that lead to a (training) set of character images are described. In Section 3, we present in detail the proposed semi-supervised clustering approach which is assisted by the user via a Graphical User Interface, while in Section 4 we describe the recognition methodology which is based on template matching. In Section 5, the experimental results are presented. Finally, conclusions and future directions are drawn in Section 6.

2. PREPROCESSING

The first stage of the proposed system consists of several preprocessing steps in order to extract possible characters of the document as connected components. The first two steps are image filtering for noise reduction and binarization followed by a connected components extraction. The main goal of this stage is to extract components of the image, which correspond to characters with high certainty. Thus, there is no need to extract all possible characters, which in practice is a challenging task. It is sufficient to extract a satisfying large set of characters in order to perform clustering on them. Later, the discarded characters can be retrieved at the recognition stage, if they correspond to a valid class, generated by the semi-supervised clustering. This approach can tolerate a more rough and simplified approximation of each step of the preprocessing stage. In order to extract a useful set of characters, capable of a meaningful clustering, a set of images, instead of only one, is preferred.

2.1 Alternating Sequential Filters

First, we apply a noise reduction technique, which is essential especially for the case of carbon-copied documents. *Alternating Sequential Filters (ASF)* [7] are selected for this purpose as they provide an effective smoothing of the image while preserving its topology, i.e. the characters shape. The ASF filtering is formulated as follows:

$$\Psi_n(g) = \beta_n(\alpha_n ... (\beta_2(\alpha_2(\beta_1(\alpha_1(g)))))),\ n = 1, 2, 3, ... \quad (1)$$

$$\alpha_r(g) = \rho^-(g \ominus rB|g), \quad (2)$$

$$\beta_r(g) = \rho^+(g \oplus rB|g) \quad (3)$$

where ρ^- is *Reconstruction Opening* and ρ^+ is *Reconstruction Closing*.

We choose to use ASF up to level 2, i.e. $n = 2$, while the initial structural element B is a disk of radius 1. The result of the ASF filtering over an image is depicted in Figure 2(b).

2.2 Image Binarization

The selected binarization method is Sauvola's approach for adaptive thresholding [8].However, as it was mentioned before, we only need to extract a satisfying number of connected components, corresponding to characters, and not a fine binarization result. A rough estimation of the foreground regions, where oversegmenting of a character binary mask is avoided, is preferable. Thus, Sauvola's parameters are fixed, using $k = 0.2$ and window size 50×50. The binarization of a selected document part is depicted in Figure 2(c).

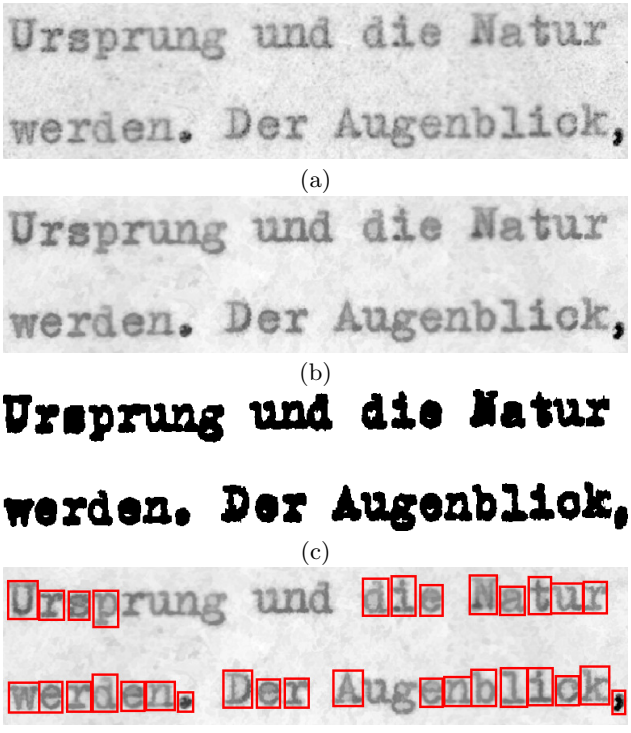


Figure 2: Summarization of the preprocessing steps. (a)Initial gray-scale part of a typewritten document (b)Result of the ASF filtering (c)Sauvola's binarization (d)Connected components selection

2.3 Connected Components Extraction

The final step of the preprocessing is the extraction of appropriate connected components that should be, most likely,

characters. Although the connected components of the binarized image will vary greatly in size, a large number of the components correspond to single characters with similar size. Under this assumption we calculate the median width w_m and height h_m of the bounding boxes of all the connected components and we assume that all the possible characters are connected components with similar size. Thus, a connected component within a specified range of bounding box width and height (related to w_m,h_m) is accepted as a character candidate. The rest connected components (noise, pictures in the documents or group of characters) are discarded.

Finally, upon finding the bounding box of the selected connected components, we extract the character candidates over the gray-scale image (after ASF filtering), as shown in Figure 2(d).The set of the extracted character candidates is the input of the next stage in order to perform the semi-supervised clustering.

3. CHARACTER CLUSTERING

In this section we will describe our approach for a user-assisted system for effective character clustering. The proposed semi-supervised clustering is performed on the feature descriptors of the character images, using Histogram of Oriented Gradients [9]. Our main focus was on the user perspective, trying to maintain a simply yet effective interaction. Specifically, we define simple actions for the user that lead to an improvement of the existing clustering. For this purpose, we developed a GUI which will be described in detail further on.

3.1 Feature Extraction

Instead of using the raw image of a character, a feature descriptor would be more convenient in terms of simplicity and speed for the upcoming clustering. Consequently for each character candidate a feature vector is extracted using Histograms of Oriented Gradients (HOG) [9]. Before using the HOG feature extraction technique, a uniform padding of the characters is applied in order to impose the same size in the set of the images. The common size is selected as the maximum possible size amongst the character candidates.

Histograms of Oriented Gradients (HOG) is a state-of-the-art feature extraction technique. HOG descriptors are widely used due to their effectiveness in recognition tasks, while retaining a simple and fast implementation. The main concept is to encode the direction of the gradient for a set of regions, which called cells, as a local histogram.The final descriptor is the concatenation of all the local histograms.

The computation of the directional gradients G_x and G_y of the image $I(x, y)$, along x-axis (horizontal) and y-axis (vertical) respectively, is performed using the following filter kernels: $[-1\,0\,1]$ and $[-1\,0\,1]^T$. In order to compute the gradient orientation at each pixel, a transformation into polar coordinates is performed through Eq. 4, 5. Only the orientation $\angle G$ is of interest and a wrapping is performed so that the orientation values lie on the interval $[0, 180^\circ)$ instead of $[0, 360^\circ)$ ("unsigned" gradient as in [9]).

$$|G(x,y)| = \sqrt{(G_x^2(x,y) + G_y^2(x,y))} \tag{4}$$

$$\angle G(x,y) = arctan\Big(\frac{G_y(x,y)}{G_x(x,y)}\Big) \tag{5}$$

Cells are extracted using a rectangular grid upon the image and within each cell a histogram of a set of orientations (bins) is constructed. For our implementation we selected a 7×5 grid, obtaining a total of 35 cells. The local histogram is computed by each pixel voting in the corresponding orientation bin with weight equal to its magnitude. The number of orientation bins is predefined and for our problem we chose 6 bins (30° range). Due to the simplicity of the problem no block-normalization is performed.

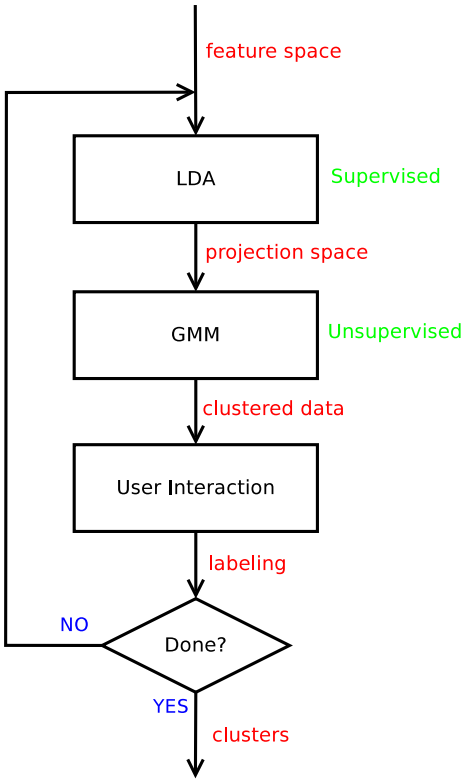


Figure 3: Overview of the proposed semi-supervised clustering approach.

3.2 Semi-Supervised Clustering

The proposed approach for the clustering of the extracted character candidates tries to minimize user interaction, assuming that if a cluster has a large number of instances of the appropriate character and relatively few other ("misclassified") characters, there is no need for further refinement. This assumption is based on the fact that the aforementioned conditions are sufficient for extracting a pattern for the character.

The proposed semi-supervised clustering consists of three main components: Linear Discriminant Analysis (LDA), Gaussian Mixture Models (GMM) and User Interaction, as it is summarized in Figure 3.

3.2.1 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) [10] is a supervised dimensionality reduction method, which projects the initial data to a subspace via a linear projection matrix. The projection is performed by optimizing the Fisher's criterion, maximizing the inter-classes variance while minimizing the intra-class variance. This is interpreted as trying to reduce the dimensionality while enhancing the separability of the

classes. The aforementioned criterion can be formulated as a generalized eigenvalues problem and, therefore, can be solved analytically by using Singular Value Decomposition. The resulting subspace has dimensionality equal to the number of classes minus one.

The use of LDA imposes the user information about the character clustering upon the projected data, incorporating a supervised step to the proposed methodology. The whole concept is in a manner similar to must-link and cannot-link constraints [5], but in a more compact way. Furthermore, the projected data consists of significantly fewer dimensions simplifying the upcoming unsupervised clustering for two reasons. First, the complexity is reduced as it is proportional to the the dimensionality of the features and, second, more meaningful clustering results are generated, avoiding the so-called dimensionality curse.

3.2.2 Gaussian Mixture Model

Instead of using k-means for the unsupervised clustering task, the more generic Gaussian Mixture Model (GMM) [10] method is preferred. In fact k-means is a deterministic special case of GMM. Specifically, each cluster in GMM is a gaussian distribution with an assigned probability. GMMs can simulate a wider variety of cluster shapes due to the covariance matrix (hyper-ellipsoid), which enables a more realistic interpretation of the data distribution, rather than using fixed-sized hyper-spheres for each cluster as k-means does.

Similar to k-means, the GMMs are generated by fitting them to the data via an iterative process which maximizes the overall likelihood of the model. The parameters of the model are estimated using the Expectation Maximization (EM) algorithm. EM finds a local maximum of the likelihoodl, which may result in varying resulting parameters for different initializations, as in k-means.

The covariance matrix inversion that is required during the EM algorithm is performed using Cholesky Decomposition in order to address the case of ill-conditioned inversion of a low-rank matrix. The aforementioned case is rather common, especially for the sub-clustering task that will be described in the upcoming section.

3.2.3 User Interaction

Despite of how effective a clustering technique is, it will not provide satisfying results and in the absence of labeled data in order to use a supervised technique, the user should provide the necessary information about the quality of the clustering or a new/different labeling. Our purpose is to make the system provide highly representative information about existing clusters and also incorporate effectively all the information the user can give in an iterative manner as in Figure 3.

The representation of each cluster is done by performing a sub-clustering. The number of sub-clusters is fixed to 4, as it is shown to lead to a fair representation of the inner structure of each class. As in general clustering, GMMs are also used for sub-clustering for the aforementioned reasons. However, GMMs are used directly to the initial feature space, rather than the projected, because the projected is biased by the previous labeling. Due to the dimension of the initial data and the reduced number of data in each cluster the generated covariance matrices, most likely, are not full-rank and therefore Cholesky Decomposition is used, as it was mentioned before.

The interaction of the user is constrained to a set of specific actions, provided by a Graphical User Interface, which are described in detail in the following section.

3.3 Graphical User Interface

The user interacts with the clustering procedure though a Graphical User Interface (GUI), which provides all the necessary actions. An example of using the GUI is depicted in Figure 4, where all the important actions and the provided information are annotated. As it is observed in Figure 4, the layout of the GUI (for a selected cluster) is simple without providing overflowing and rather unnecessary details. The provided user-friendly actions and representation of the cluster, which will be described in detail below, encourage the user in contrast to the rather bothersome procedure of "carpet"-like approaches [6].

3.3.1 Sub-Clustering

As mentioned before, a rough estimation of the inner structure of the class is given by sub-clustering each generated cluster. The GUI provides a mean image as a representative for every sub-cluster, as is shown in Figure 4. The mean images should be incomprehensible if many instances of different characters belong to the same sub-cluster, but will be rather clear if the majority of the instances comes from the same character. For clarity, the number of the associated character images is displayed for each sub-cluster in the top-left corner of the representative image, as it is marked in Figure 4. The purpose of providing the number of images in each sub-cluster is to measure the "strength" of the sub-cluster, i.e. how much does it contribute to the whole cluster.

3.3.2 Actions

The user can navigate through the existing classes and for each sub-cluster, four actions are possible:

- **None** No action. The sub-cluster is in correspondence with the current cluster and therefore no change is needed.
- **Junk** Delete every descriptor in this sub-cluster (mostly for extremely noisy set of images or over-segmented components of characters). This action is used for discarding a set of images that does not correspond to a valid character and may affect the clustering. A representative example is presented in Figure 5(a).
- **Unassign** Unassign the descriptors and expect a more satisfying clustering in a next iteration. It is possible that a sub-cluster consists of instances of more than on character and thus a reassignment of its contents is needed. This can be accomplished in the next iteration of the proposed system, after registering the user's actions. However, we do not want this "miss-labeling" to contribute to the supervised LDA, but only in the upcoming clustering. In other words, we want to keep this sub-cluster "hidden" in the LDA step. This is exactly what the "unassign" action does. An example of a sub-cluster consisting of the characters 'T' and '2' and should be unassigned is shown in Figure 5(b).
- **New Cluster** Consider the sub-cluster as a new cluster, thus assigning it a label. If the label already exists

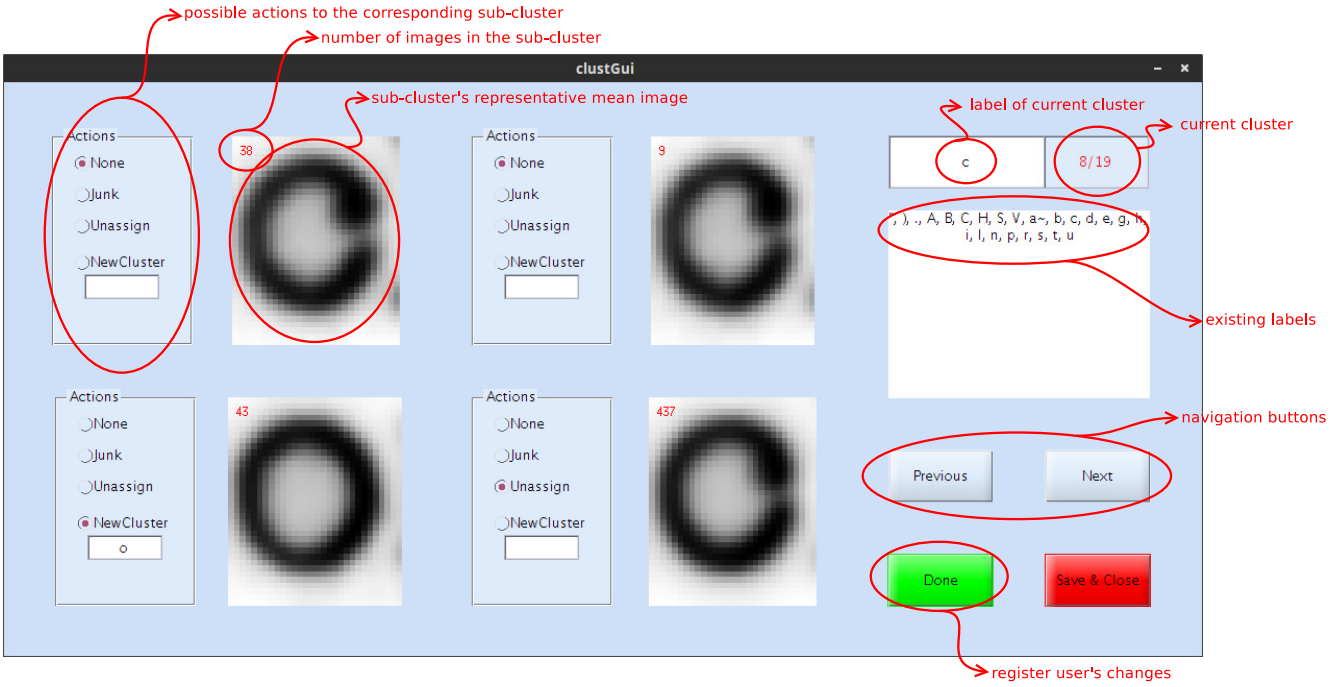
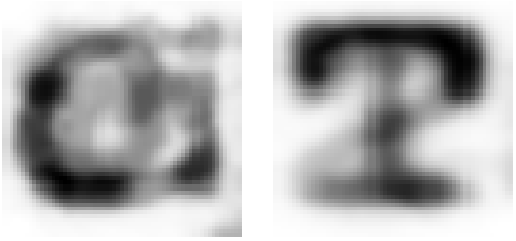


Figure 4: Graphical User Interface.

a merge is performed between the current sub-cluster and the appropriate cluster. This action is performed when the sub-cluster is clearly a different character from the corresponding cluster's character. Such cases are common when the number of labeled clusters is smaller than the different existing classes/characters, so the user can help the system "discover" new character classes or correct a bad clustering result.



(a) (b)

Figure 5: Example of the representative mean image for the case (a) *junk* action (b) *unassign* action

Summarizing, given the representative mean images of the sub-cluster, the user should decide which action is the most appropriate one for a better clustering and eventually unvarying sub-clusters. Such a scenario is depicted in Figure 4, where in the cluster 'c' are also instances of the character 'o' ('o' hasn't been "discovered" yet). So we choose the new-cluster action for the bottom-left sub-cluster, while assigning the label 'o', and the unassign action for the bottom-right sub-cluster, which seems to consist of both characters.

It should be noted that the user should identify each constructed cluster and mark it with the corresponding label. This labeling is essential for the upcoming recognition stage.

3.3.3 *Cluster Initialization*

For the initialization of the clustering procedure the GUI provides two options: 1) a k-means clustering on the initial feature descriptors to a predefined number of clusters and 2) the usage of a clustering result from a previous use of the GUI. The second option apparently speed up the procedure and can be used to quickly adapt (with the least effort by the user) to slightly different font for a different set of images.

4. OPTICAL CHARACTER RECOGNITION

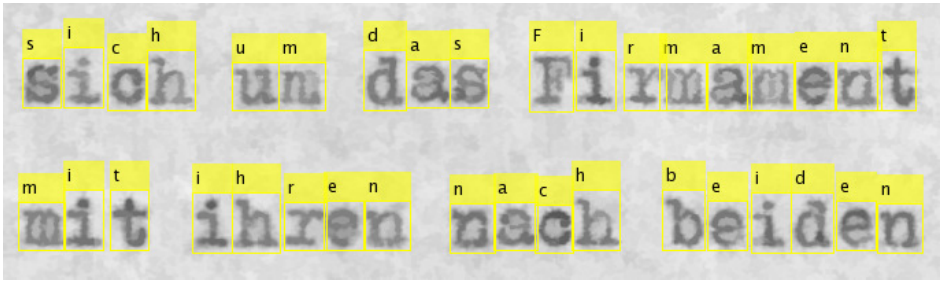
The final stage of the proposed system is the character recognition according to the generated character classes. This procedure is in fact independent of the previous steps, i.e. given the generated classes from a set of images the recognition scheme can be applied to a different set. The only prerequisite is that the training and the testing sets have the same font (a possible difference in scale can be easily computed using the median connected component size, as in the preprocessing stage). Additionally, similar to the preprocessing stage, an ASF filtering is applied to the images before recognition.

4.1 Template Matching

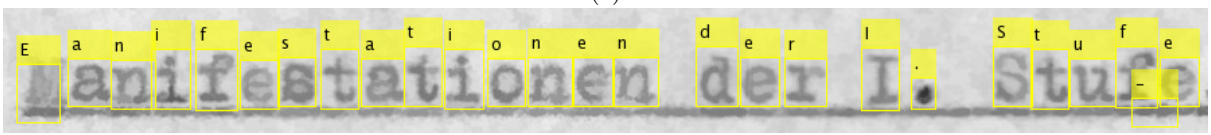
Recognition is performed using template matching (a template corresponds to a generated class), thus providing a segmentation-free OCR methodology. The choice of template matching is related to our approach of constructing an effective system consisting of simple components. However, contrary to the rest of the system components, template matching is rather slow due to the convolution of each template with the document image.

4.1.1 *Template Selection*

First, a template for each cluster is generated. We ob-



(a)



(b)

Figure 6: Examples of Optical Character Recognition.

served that for a sufficiently good clustering, which is proportional to the time devoted by the user during the semi-supervised clustering, a mean image of the raw character images is a valid choice. The mean image, given that it is generated from a sufficiently large set of images, is representative of the class, while noise is significantly reduced.

4.1.2 Normalized Cross-Correlation

Having extracted the representative templates, the template matching is performed using normalized cross - correlation which is computed for every pixel via Eq.6 (μ is the mean, σ is the standard deviation and W is the set of pixels corresponding to a window equal to the template's size). Normalized cross-correlation, denoted by c, takes values in the range $[-1, 1]$, where 1 denotes an exact match and -1 denotes complete dissimilarity. It is a widely used technique for template matching due to its contrast insensitive property; a useful property for detecting faint characters. This insensitivity is achieved by dividing the convolution with the local standard deviation of the image and the template's standard deviation. The normalized cross-correlation is implemented according to [11], using integral images for the computation of the local mean and deviation of the document image.

$$c(u,v) = \frac{1}{|W|} \frac{\sum_{x,y \in W} [f(x,y) - \mu_f][t(x-u, y-v) - \mu_t]}{\sigma_f \sigma_t} \tag{6}$$

However, the aforementioned insensitivity is a hindrance in some cases, resulting to false detections, e.g. noise with high variance. Therefore, a semi-normalized approach is adopted. Specifically, we assign a threshold in standard deviation σ_{th}, such that the local deviation is constrained as given by Eq. 7.

$$\sigma'_f(x,y) = \begin{cases} \sigma_f(x,y), & if\ \sigma_f(x,y) > \sigma_{th} \\ \sigma_{th}, & if\ \sigma_f(x,y) \leq \sigma_{th} \end{cases} \tag{7}$$

We chose this threshold to be twice the median[1]of the local standard deviation of every background pixel, which is denoted by σ_b, i.e. $\sigma_{th} = 2\sigma_b$. We can find the median standard deviation of the background using Sauvola's binarization method with almost no extra computational cost, as the local mean and standard deviation have already been computed.

4.1.3 Character Recognition

After performing the template matching, we obtain the normalized cross-correlation coefficients for every pixel and we want to find the possible positions/regions of the recognized character. In order to obtain a reasonable number of possible candidates for each template we discard locations where $c(x,y) < c_{th}$. Furthermore, we retain only the positions corresponding to local maxima by performing a dilation on c with a structural element of the size of the template. The threshold c_{th}, after experimentation, is set to 0.75 which assumes a fair similarity between the template and the corresponding region of the image.

Concerning the final recognition of the image characters, the issue of overlapping regions between different templates should be addressed. Generally, the best spatial arrangement for a set of point can be found by dynamic programming as a generalization of the best sequence problem (e.g. Viterbi algorithm). However, for our simplified case, where usually very small groups of character are overlapping significantly, this can be resolved by finding such each group and choose only a candidate template as a "winner". The winner is the character with the biggest similarity value. The results of the character recognition can be shown in Figure 6. It should be noted that the presented segmentation-free method works well even for underlined text as it can be observed in Figure 6(b).

4.2 Page Recognition

The last step of the recognition stage, in the context of a complete OCR system, is the retrieval of the text from the document image. Given the labels of the recognized characters and their positions, our final task is to align them into a readable text. This task is performed in a similar way to simple line detection techniques (e.g. using Hough transform) under the assumption that lines can be easily separated, which is a fair assumption for typewritten documents.

[1]the choice of median instead of mean is made in order to obtain a less sensitive value regarding outliers.

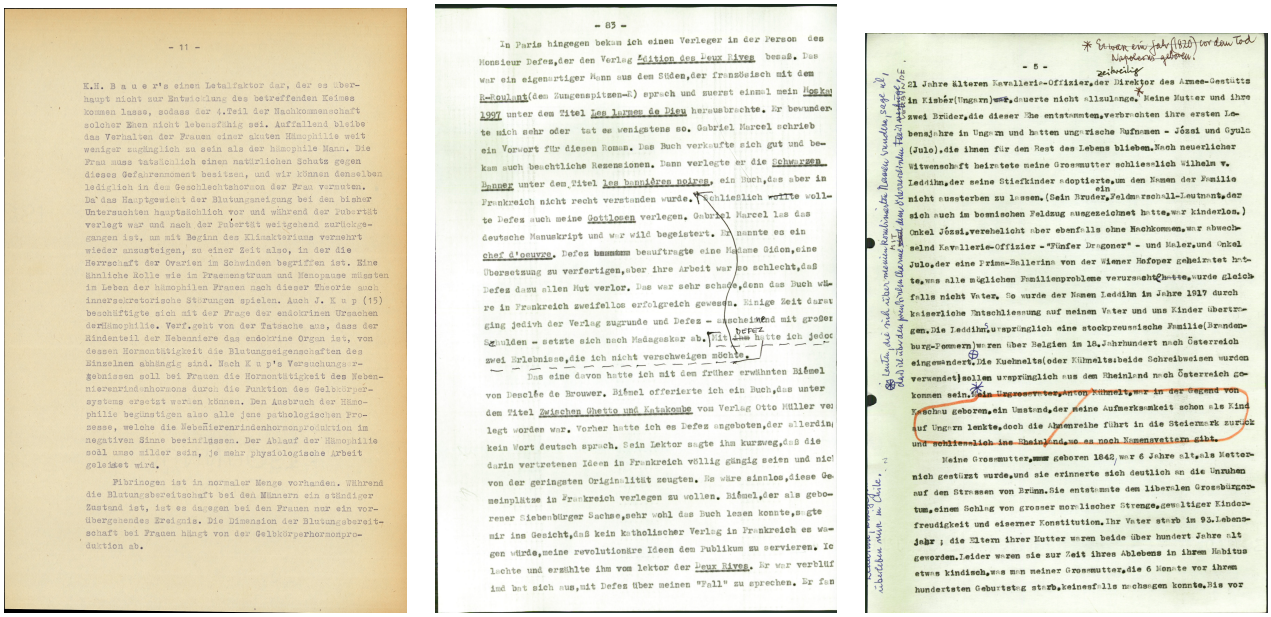


Figure 7: Examples of Dataset Images.

5. EXPERIMENTAL RESULTS

Having described the proposed system, we want to evaluate it in terms of recognition on a selected dataset. Recognition results serve as an indirect evaluation of the quality of the generated classes by the proposed clustering approach.

The dataset contains typewritten pages of technical/ scientific nature from the collection of the University of Innsbruck (in German). Some pages are originals (directly impacted by the typewriter keys and ink ribbon) and some are carbon copies (produced by the force of the typewriter keys through a carbon sheet behind the originals). There are occurrences of manual corrections and annotations in various places. All pages were scanned at 300dpi, the majority in color - the rest in grayscale. Some examples of documents featuring the variety of the dataset are presented in Figure 7. The database consists of 8 different sets of document pages, which may vary in font or level of noise, i.e. original or carbon copy.

The number of pages that are used for feature extraction and clustering are 10 and correspond to the training set. We have observed that in practice the choice of 10 pages is sufficient and produces $\sim$ 15000 character images. An increase in the number of training image doesn't have any significant effect in the accuracy of the proposed system and slows noticeably the procedure, making the use of the GUI time-consuming due to unnecessary waiting between each iteration.

The evaluation of the system is performed by measuring the error rate of the recognized text, defined as the edit distance between the generated text and the groundtruth divided by the number of the characters in the groundtruth text. The spaces are discarded from every text in order to have a meaningful error rate since the proposed system does not count them (there isn't a template corresponding to space).

Our proposed methodology for OCR is compared to the ABBYY FineReader11.0 [12], which is set to typewritten German in order to be consistent with the used database.

Initially we choose only one set and perform the proposed clustering scheme with the aforementioned GUI on 10 randomly chosen pages from this set. The initialization of the GUI clusters is performed using k-means clustering, where $k = 10$. We evaluate our OCR for the following cases: 1) the remaining set ($SetA$) 2) the rest of the sets ($SetB$). This is done in order to evaluate the dependency of the OCR to the font and the variability of the different sets. The resulting error rates are shown in Table 1. It is observed that the recognition error of the proposed system is significantly lower for the set used for training, mainly due to the same font, outperforming FineReader. However, in the more generic case of SetB, the Finereader preserves a similar error rate in contrast to our methodology which deteriorates noticeably. The main reason for that is the variability of the fonts in the sets. Considering user's interaction the whole procedure takes about 15 minutes with a random k-means initialization, which assumes no prior knowledge. This can be considered relatively fast, considering that the user assisted the system to "discover" as much existing characters as possible without any initial information.

Table 1: Error Rate for Page Recognition

	FineReader11	Proposed System
SetA	12.20%	6.00%
SetB	10.92%	16.38%

To overcome the aforementioned problem, we try to adapt the clustering results to each set using the GUI initialized to a previous clustering in order to speed-up the procedure. Therefore we introduce a second scenario where having as initial clustering the one from the first experiment, we retrain our system to the current set (by choosing 10 training images). Using the aforementioned initialization, the user's interaction time is significantly reduced to about 5 minutes,

Table 2: Error Rate for Page Recognition after Retraining each Set

	FineReader	Proposed System	Retrained Proposed System	Unidentified Characters (#Clusters)
Set1	12.20%	6.00%	6.00%	0.65%(63)
Set2	15.77%	23.70%	18.90%	13.43%(62)
Set3	15.75%	16.18%	8.51%	1.53%(65)
Set4	9.62%	16.15%	11.27%	1.16%(67)
Set5	7.62%	18.51%	12.04%	1.31%(66)
Set6	7.84%	15.06%	10.43%	1.31%(69)
Set7	7.58%	15.73%	9.56%	1.28%(69)
Set8	4.41%	9.85%	6.17%	1.63%(64)
Total	10.92%	16.65%	11.62%	

since most of the clusters are found correctly instantly and only minor changes are required. The recognition results are presented in Table 2 along with the error rate without re-training (corresponding to the first experiment) and the percentage of unidentified characters of the groundtruth. As unidentified characters we refer to those which do not correspond to any of the generated classes (the total number of classes that our system produced in each set is also provided in parenthesis in Table 2). The overall recognition rate of our system is greatly improved, as expected. Individually for each set, the proposed system outperforms FineReader only for two cases.

The main reason for the increased error rate in specific sets is the state of the documents, where many faint characters are present and template matching is not very suitable for these cases. Additionally, a reason which leads to a further deterioration of the recognition result is that the recognized text consists of as many character classes as the generated clusters, while the groundtruth text may have more classes (either they are not present in the training images or the system was unable to "discover" them mainly due to their few occurrences). This can be observed at the last column of Table 2. It should be noted that a simple template-matching technique provides promising results after a brief period of user's assistance, even though no pre-trained models are used (e.g. FineReader).

6. CONCLUSIONS

A trainable historical typewritten document recognition system is presented that focuses on a simple but yet effective user interaction. Specifically, the user provides the necessary feedback, using simple predefined actions via the presented GUI, for the realization of an efficient semi-supervised clustering system. The extracted clusters are used to generate templates and a template matching is performed, resulting to a segmentation-free optical character recognition methodology. The proposed system has efficiently clustered the classes/characters for historical typewritten documents and obtained low error rate in the OCR task, especially after performing a fast re-clustering in order to capture changes in the font. A possible future extension of the presented work will be the substitution of the recognition step, i.e. the template matching, with a more sophisticated methodology. Specifically, state-of-the-art classifiers can be utilized while working on a feature space, such as the HOG descriptors or even the projected space generated by the LDA technique during the proposed clustering approach, rather than on raw images.

7. ACKNOWLEDGMENTS

The authors would like to express their gratitude to Dr Günter Mühlberger of the University of Innsbruck for making the dataset available to them for research purposes.

8. REFERENCES

[1] M. Cannon, J. Hochberg and P. Kelly. Quarc: A remarkably effective method for increasing the ocr accuracy of degraded typewritten documents. *Proc. 1999 Symp. on Document Image Understanding Technology*, pages 154–158, 1999.

[2] A. Antonacopoulos and D. Karatzas. Semantics-based content extraction in typewritten historical documents. *8th Int. Conf. on Document Analysis and Recognition*, pages 48–53, 2005.

[3] A. Antonacopoulos and C. Casado Castilla. Flexible text recovery from degraded typewritten historical documents. *Proc. 18th Int. Conf. on Pattern Recognition*, pages 1062–1065, 2006.

[4] G.P. Silva and R.D. Lins. An automatic method for enhancing character recognition in degraded historical documents. *Proc. 11th International Conference on Document Analysis and Recognition*, pages 553–557, 2011.

[5] S. Pletschacher, J. Hu and A. Antonacopoulos. A new framework for recognition of heavily degraded characters in historical typewritten documents based on semi-supervised clustering. *Proc. 10th International Conference on Document Analysis and Recognition*, pages 506–510, 2009.

[6] C. Neudecker and A. Tzadok. User collaboration for improving access to historical texts. *Liber Quarterly*, 2(1):119–128, 2010.

[7] S.R. Sternberg. Grayscale morphology. *Computer Vision, Graphics, and Image Understanding*, 35:333–355, 1986.

[8] J. Sauvola and M. Pietikainen. Adaptive document image binarization. *Pattern Recognition*, 33(2):224–236, 2000.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *Proc. IEEE Conference Computer Vision and Pattern Recognition*, 2:886–893, 2005.

[10] R.O. Duda, P.E. Hart and D.G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[11] J.P. Lewis. Fast normalized cross-correlation. *Industrial Light & Magic*, 2000.

[12] ABBYY FineReader 11.0: http://www.abbyy.com.