A Self-Adaptive Method for Extraction of Document-Specific Alphabets

Stefan Pletschacher
Pattern Recognition and Image Analysis (PRImA) Research Lab
School of Computing, Science and Engineering, University of Salford, Greater Manchester, United Kingdom
s.pletschacher@primaresearch.org

Abstract

Recognition and encoding of digitized historical documents is still a challenging and difficult task. A major problem is the occurrence of unknown glyphs and symbols which might not even exist in modern alphabets. Current pre-trained OCR-methods hardly deliver usable results for such documents. This paper describes an alternative approach and framework for handling printed historical documents without restrictions on the contained alphabets or fonts. The basic idea is to derive all information required for encoding directly from the document itself. This is achieved by extracting a document-specific prototype alphabet of locatable glyphs. Core of the system is a customized clustering method which adapts automatically to new documents by ascertaining appropriate threshold parameters based on the special characteristics of glyphs. This way, the system is able to run without manual interventions and can be integrated into automated mass digitization workflows.

1. Introduction

The efforts to digitize historical documents as part of our cultural heritage have dramatically increased over the last years. Ultimate goal for these resources is completely recognized and hence searchable text. Accordingly, research into improved and more specialized Optical Character Recognition (OCR) currently constitutes an important field. However, there is also a huge amount of documents which will probably not benefit from those endeavors. This is especially true for documents containing ancient scripts, obsolete typefaces as well as unknown characters and symbols which cannot be handled by pre-trained OCR-engines. Figure 1 shows a typical example of a historical document which cannot be completely recognized or encoded using conventional OCR due to the inherent alphabet. Many of regularly used glyphs in historical documents have no counterpart in modern, computer-based alphabets.

As manual transcription of such documents is not applicable to mass digitization, it is desirable to have alternative means for automated processing without any restrictions on the contained alphabets, scripts or fonts. Moreover, for the representation of digitized documents as close to the original as possible, it is also required to avoid any alterations of the material. It is therefore important to have access not only to contained text of source documents, but also to their original appearance. This includes not only the reproduction of original fonts and special symbols but also broken characters or mistakes in writing (which conventional dictionary based OCR-methods wrongly tend to correct).

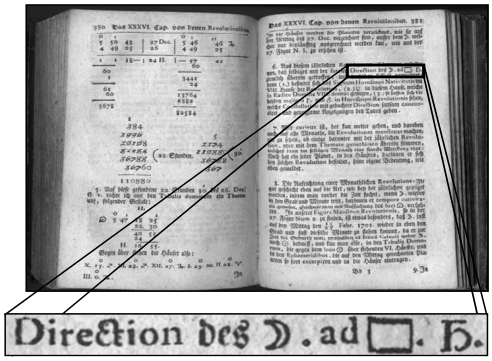

Figure 1. Examples for ancient glyphs which are not part of modern alphabets.

In the next section, a framework for handling printed documents which are not suitable for current OCR-methods is presented. Subsequently, the background and development of a module for the extraction of document-specific alphabets is described, followed by a discussion of particular effects and concluding remarks.

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.253

2. OCR-independent document encoding

Conventional OCR-methods rely on known features or instances of target characters for good recognition results. Together with appropriate dictionaries for post correction they are able to achieve acceptable recognition rates. Unfortunately, this background knowledge is very specific to different document types and requires a high effort to be obtained. Optimization of OCR-engines to support a broader variety of documents is absolutely desirable. However, for a lot of unique material like the numerous historical documents in libraries such an effort cannot be justified if only few instances benefit. As a consequence, such material must be either manually transcribed or left untouched unless a new approach without OCR can be provided.

The main idea at this point is not to rely on any prior knowledge or predefined alphabets but to extract all necessary information directly from the documents themselves. For putting this into practice, a document-specific alphabet has to be extracted from the original. This can be achieved by clustering all occurrences of characters into classes and then representing each class by just one prototype [1]. Once this document-specific alphabet has been determined, the original may be encoded by replacing all individual characters with a reference to their particular prototype. This follows the basic idea of symbolic compression [2].

Only few systems are pursuing this general approach (e.g. [3]). The work presented here, however, goes much further by integrating the ability to adapt to new document types automatically and by generating a corresponding vector font along with the document-specific alphabet. To obtain such a generic font, each prototype is transformed into a path description using a specialized raster-to-vector conversion which preserves the appearance of glyphs as close to the original as possible. Intermediate result is an SVG-font which can be directly integrated in XML-based descriptions of encoded documents [4]. Suchlike represented documents, based on a document-specific alphabet and font, can be easily processed, transformed and output on various platforms using standard XML-technologies.

3. A framework for encoding and repurposing of digitized documents

The outlined extraction of document-specific alphabets is part of a larger encoding and repurposing framework [5]. This system consists of several coupled modules. Namely, the modules include preprocessing, text segmentation, alphabet extraction, font generation, document encoding, and a repository which also comprises means for repurposing and transforming encoded documents into several output formats.

The framework operates as follows: Scanned facsimiles of originals are fed into the repository and registered. Once all document pages are available, the preprocessing step can be initiated in order to obtain text blocks which are relevant for further preparation. Other document parts like figures or graphics are dealt with separately. Text segmentation detects then the logical structure of lines, words, and characters and delivers individual images for all glyphs. Those are further analyzed for ascertaining the prototypes of a document-specific alphabet. Subsequently a vector font is created by applying a customized vectorization method to all prototype images. Code points from the Unicode private use areas are eventually assigned to all prototypes to be used in the encoded representation of the originals. The encoding module stores this individual alphabet together with the vector font in the definition section of the output XML instances. The actual content is then composed of references to the respective code points. Specific repurposing (e.g. text reflowing, accessibility improvement etc.) and transformation methods can now operate on this self-contained document representation and produce output formats for various distribution channels like web, print-on-demand or mobile viewing devices.

4. Self-adaptive extraction of document-specific alphabets

Core part of the aforementioned framework is the alphabet extraction module as it determines the unique glyph prototypes inherent in the present document. Main task of this module is hence to calculate an optimal clustering of all glyphs found in the original and to ascertain one representative for each cluster. In order to allow a high degree of automation, the method has to estimate proper process parameters (e.g. the similarity threshold to distinguish glyphs during the clustering) automatically rather than asking for input from a human operator. Moreover, it is also desired that the method adjusts to new document types automatically. This is necessary as the system is intended to handle various kinds of scripts, languages and fonts which may all require individual settings. So far, optimal parameters can only be detected afterwards, comparing actual clustering results with ground-truth for all different configurations. Unfortunately, this ground-truth does not exist for unprocessed documents, which prevents direct

parameter evaluation. The method outlined in the following overcomes this limitation by introducing an auxiliary metrics for this purpose.

4.1. Clustering

For the specific task of calculating a document-specific glyph alphabet there are two major constraints on possible clustering algorithms:

1. The number of true classes is unknown. This results directly from the approach not to limit the set of processible characters in any way, in contrast to conventional OCR-methods which rely on predefined alphabets.

2. The number of glyph instances to be processed is potentially very high. This is especially true for the envisaged application to mass digitization of whole books which may contain up to millions of glyphs.

Accordingly, methods neither requiring the number of clusters as input nor comparing each glyph to each other, resulting in exponential runtime, are applicable. The implemented method is therefore based on adaptive vector quantization [6, 7], which works iteratively and dynamically creates new clusters starting from an empty codebook (set of prototypes).

The algorithm works as follows: The first incoming glyph forms the first cluster and prototype. The algorithm then runs over all remaining glyphs and creates a new cluster whenever the maximum similarity of the currently processed glyph, compared to all existing cluster prototypes, lies below a specified threshold. Otherwise, the present glyph is merged into that cluster with the highest similarity value. Every time a new glyph is merged into an existing cluster the particular prototype representing this class needs to be updated. This causes prototypes to slightly change with a growing number of contained glyphs. The actual prototype bitmap is determined based on the density of each pixel contributed by all represented glyphs. Hence, distortions and noise occurring only in few glyphs are effectively suppressed.

The required similarity measure is calculated by means of a weighted pattern matching operating on the glyph images. In order to reflect glyph characteristics more precisely it takes into account both, error accumulations and the distance of differing pixels to the main component by applying corresponding penalties. This is especially necessary to discern minor details with a high impact on the accuracy (like the separation of "O"s and "Q"s). Feature based similarity measures turned out to be less effective. This is due to the followed universal approach of finding glyphs with very similar visual appearance in contrast to omni-font recognition which usually benefits from feature based methods.

4.2. Adaptivity

In order to enable automated workflows, the alphabet extraction module requires the ability to ascertain the aforementioned optimal threshold for the similarity measure automatically, depending on the actual input. Optimal means in this context to fulfill two conditions:

1. Avoid misclassifications in form of substitution errors (e.g. a glyph of true class "A" wrongly assigned to a cluster containing only glyphs of true class "B"). This is necessary to achieve a low error rate (ratio of substitution errors to total number of glyphs) and hence, to prevent falsification of the content.

2. Minimize the number of obtained clusters (down to the true number of classes in the original) to obtain a high compression rate (ratio of found clusters to true classes) in terms of symbolic compression.

Unfortunately, measures to optimize one of the two tend to have a negative effect on the other. This trade-off has to be considered when identifying the optimal threshold. Since a major goal of the framework is robustness, a low error rate must be given priority over an optimized compression rate.

Given a labeled set of glyph images (i.e. ground-truthed) the detection of the optimal threshold is a straightforward task. The corresponding algorithm loops over all possible thresholds and performs the particular clustering, calculating substitution error rate and compression ratio. The optimal value can then be selected considering the above two conditions.

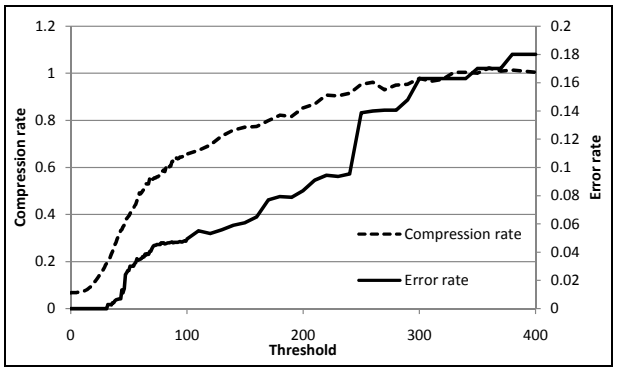

Figure 2. Error and compression rate for increasing threshold (Comp. rate 0 – no comp.; 1 – maximum symbolic comp.; greater 1 – lossy compression).

Obviously, the compression rate increases with higher thresholds. Upper limit is the case when all

glyphs end up in one single cluster. The other extreme is a zero error rate in terms of substitutions, however, with no compression at all. This is the case when each glyph forms a single cluster. The optimal threshold is hence the highest one which still does not cause substitution errors. Figure 2 shows the typical behavior of error and compression, obtained from a digitized and ground-truthed historical document containing German Fraktur.

The function of substitution errors over all thresholds is unfortunately not available for new input material and can only be obtained via labor extensive, manual ground-truthing. As this is not feasible in automated workflows, it is desirable to find a function which a) can be calculated directly from the input and b) shows the same or at least a similar behavior like the error function.

Experimental results show that the maximum intra cluster distance can be used for this purpose. It constitutes the maximum difference of the similarity measure between any glyph and its cluster prototype among all clusters. This function can be calculated based solely on the clustering results for all considered thresholds. Moreover, a strong correlation between the maximum intra cluster distance and the error rate can be shown for various document types. Figure 3 contains the correlation coefficients from experiments with different scripts and fonts.

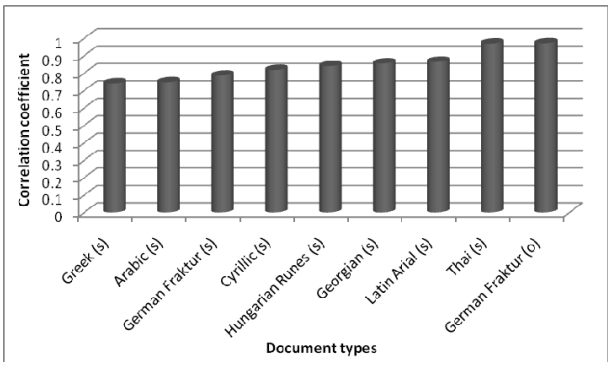

Figure 3. Correlation between maximum intra cluster distance and error rate for different document types; (s) – synthetically created glyphs; (o) – glyphs from a digitized original.

Values close to 1 indicate a high linear correlation i.e. potential dependency between the two functions. Due to the high effort for manually producing ground-truthed test material, synthetic glyph sets have been created in addition to genuine glyph images from digitized originals. This was achieved by rendering characters with known labels using TrueType fonts and subsequently applying artificial deterioration.

The rationale for the correlation and actually true dependency is that an increasing threshold causes bigger clusters which contain also less similar glyphs (i.e. with a higher distance to the prototype). This goes along with the introduction of misclassifications and hence the error rate. The maximum distance shows this effect better than the average or minimum distance since it reflects the least similar elements of clusters, i.e. the ones which are most likely to introduce errors.

In terms of finding the optimal threshold, the task is now to find the point from where errors are introduced. The observation from ground-truth based experiments is that both, error rate and maximum intra cluster distance, show a leap when the threshold becomes too high and misclassifications start.

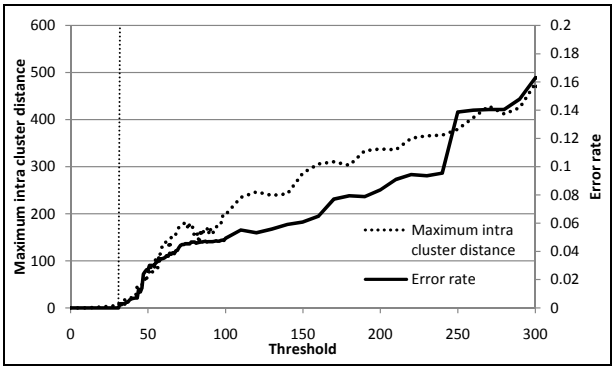

Figure 4. Selection of a safe threshold based on the maximum intra cluster distance.

The threshold selection method must choose a safe value in order to guarantee stable and robust processes according to the aforementioned condition 1. This is achieved by approximating the first leap with a slope analysis of the function, selecting the threshold at the point where the function starts to rise. Once the threshold is found, the appropriate clustering can be determined and accordingly the set of prototypes i.e. the document-specific alphabet can be generated.

5. Discussion

The approach of the framework is built upon the assumption that self-contained documents like whole books are based on a finite number of characters and symbols which form the underlying alphabet. This alphabet is not always known in advance, especially not when dealing with historical documents. Using conventional systems, recognition and encoding of such originals is therefore only feasible for fragments or, in the worst case, entirely impossible. The presented method overcomes this limitation. Moreover, it allows completely automated workflows which are

necessary for mass digitization by automatically adapting to new document types. Encoding of digitized documents using a document-specific alphabet is also a step towards true recognition. This way, the necessary effort for manual transcription is reduced to labeling only a dramatically smaller set of prototypes. Even if this manual labeling is not carried out, there are still all the benefits from the encoded document representation (e.g. compression for storage, possibility of repurposing, cross-media publishing etc.).

The extraction of document-specific alphabets using pattern matching is feasible if the employed similarity measure emphasizes discriminating details of glyphs. Such details are inherent to any font or graphical alphabet representation since human readers must be able to distinguish individual characters. Moreover, it is valid to treat significantly different appearances of the same character (e.g. scaled or rotated glyphs) as individual classes. Identical glyphs with different meanings, on the other hand, are very rare and can only be recognized using additional context information e.g. within a post-correction module. The impact of clustering identical glyphs with different meanings into the same class is very little in the scope of the described framework. This is due to the fact that any graphical reproduction of the encoded original will again lead to a human readable form. Consequences for text search are also minimal as fuzzy query matching methods can easily deal with minor ambiguities of that kind.

The estimation of the optimal threshold, to judge about the similarity of two glyphs, plays a central role for the automation of the whole framework. There is a tradeoff between a low error rate using restrictive thresholds and a high compression rate at high thresholds. The approach of a flexible, document-specific alphabet allows prioritizing accuracy rather than compression to make systems more robust. This is possible due to the different types of potential mis-classifications. The first type occurs when a glyph is merged into a wrong cluster. This is a fatal substitution error and would cause mistakes in the encoded document. The second is less severe and arises when a glyph is erroneously put into a new cluster instead of being merged into its already existing true class. The first error type is to be avoided by all means for robust systems. The second, however, causes only redundancy (i.e. a non-optimal compression) and can be tolerated up to a certain degree. Thus, it is possible to implement the aforementioned two conditions for the optimal threshold in order to calculate the particularly best value for a given input.

6. Conclusion and future work

The functionality and background of a new method for self-adaptive extraction of document-specific alphabets has been shown in this paper. This method constitutes one of the core components of an alternative framework for document encoding independent of conventional OCR and hence not relying on any prior knowledge about underlying languages, scripts or fonts. Due to its ability to adapt to new types of documents it is well suited for integration into automated mass digitization workflows.

Future work will include further improvements of the threshold estimation method. This step is currently very time-consuming as the algorithm iterates over all possible thresholds and over all input glyphs to find the best value. It is expected to reduce the number of necessary iterations dramatically by implementing heuristics for selecting a representative glyph subset to operate on. In terms of the framework it is planned to exploit results of the alphabet extraction module to refine the preceding glyph segmentation by implementing a feedback function. Moreover, search functionalities for document-specific encoded documents will be developed. The main problem here is rather the interface for users to enter queries using a document-specific alphabet than the actual text matching.

References

[1] G. Kopec, M. Lomelin, “Document-Specific Character Template Estimation”, Proc. SPIE Vol. 2660, Document Recognition III, 1996

[2] R. N. Ascher, G. Nagy, “A Means for Achieving a High Degree of Compaction on Scan-Digitized Printed Text”, IEEE Transactions on Computers, Volume C-23, Issue 11, 1974

[3] T. M. Breuel, W. Janssen, K. Popat and H. Baird, “Paper to PDA”, Proc. International Conference on Pattern Recognition, Quebec, Canada, 2002

[4] S. Pletschacher, M. Eckert, A. C. Huebler, “Vectorization of Glyphs and Their Representation in SVG for XML-Based Processing”, Proc. International Conference on Electronic Publishing, Bansko, Bulgaria, 2006

[5] S. Pletschacher, “Representation of Digitized Documents Using Document Specific Alphabets and Fonts”, Society For Imaging Science and Technology (IS&T) Archiving 2008. Bern, Switzerland, 2008, pp. 198-202

[6] R. M. Gray, “Vector quantization”, IEEE ASSP Magazine, April 1984, pp. 4-29.

[7] J. E. Fowler, “A Survey of Adaptive Vector Quantization-Part I: A Unifying Structure”, IPS Lab. Tech. Rep. TR-97-01, The Ohio State Univ., Mar. 1997.