

Active Learning for Classifying Template Matches in Historical Maps

Benedikt Budig^(✉) and Thomas C. van Dijk

Chair of Computer Science I, Universität Würzburg, Würzburg, Germany
{benedikt.budig,thomas.van.dijk}@uni-wuerzburg.de

Abstract. Historical maps are important sources of information for scholars of various disciplines. Many libraries are digitising their map collections as bitmap images, but for these collections to be most useful, there is a need for searchable metadata. Due to the heterogeneity of the images, metadata are mostly extracted by hand—if at all: many collections are so large that anything more than the most rudimentary metadata would require an infeasible amount of manual effort. We propose an active-learning approach to one of the practical problems in automatic metadata extraction from historical maps: locating occurrences of image elements such as text or place markers. For that, we combine template matching (to locate possible occurrences) with active learning (to efficiently determine a classification). Using this approach, we design a human computer interaction in which large numbers of elements on a map can be located reliably using little user effort. We experimentally demonstrate the effectiveness of this approach on real-world data.

Keywords: Active learning · Threshold detection · Human computer interaction · Template matching · Historical maps · Knowledge discovery

1 Introduction

In this paper we apply proper data mining techniques to a problem in the digital humanities. Many (university) libraries and archives have an extensive collection of historical maps. Besides being valuable historical objects, these maps are an important source of information for researchers in various scientific disciplines. This ranges from the actual history of cartography to general history, as well as the geographic and social sciences. To give a non-trivial example: onomastics, the study of the origin and history of proper names, makes extensive use of historical maps.

With the progressing digitisation of libraries and archives, these maps become more easily available to a larger number of scholars. A basic level of digitisation consists of scanned bitmap images, tagged with some basic bibliographic information such as title, author and year of production. In order to make the maps searchable in more useful ways, further metadata describing the contained information is desirable. A particularly useful class of metadata is a *georeferenced*



Fig. 1. Place markers and text on several historical maps from the Franconica collection. Note the variety of visual styles, both in the pictographs and the lettering.

index of the contained geographical features (such as labeled cities and rivers) and geopolitical features (such as political or administrative borders). In this context, a *georeferenced* map element is one that is associated with a real-world, geographical location (in some coordinate reference system). This enables queries that are useful for actual research practice, such as “all 17th century maps that include the surroundings of modern-day Würzburg,” or comparing the evolution of place-name orthography in different regions. It also enables analyses of the geographic/geodetic accuracy or distortion of the maps, which is of historical and cartographic interest.



Unfortunately, analysing the contents of historical maps is a complex and time-consuming process. For the most part, this information extraction task is performed manually by experts—if at all. For example, it currently takes the Würzburg University Library between 15 and 30 hours to georeference just the labeled settlements in a typical map from their collection.¹ To see why it takes so long, consider that the number of labeled place markers in a map can be in the order of several thousand.

Automated tools for this task are scarce, for a variety of reasons. For one, there is a large variety of drawing styles in historical maps. This makes it hard for a single algorithm or software tool to automatically perform well on a large set of maps: see Fig. 1 for some examples of the range of styles that occur in the Franconica collection.² Secondly, there is the question of input. When an historian georeferences a map, he or she brings a wealth of background information and the ability to do additional research when required. Finally, there is the issue of correctness: in general, algorithms for extracting semantic information from bitmap images are far from perfect. This is to be expected since these problems are truly difficult for computers. To the curators of historical map collections, however, the correctness of metadata can be of paramount importance (not to mention: a matter of pride).

In light of the above difficulties, we have developed an active-learning system for a generally-applicable subproblem in this area. In this paper we demonstrate

¹ Personal communication with Dr. H.-G. Schmidt, head of the Manuscripts and Early Prints department, Würzburg University Library.

² Würzburg University Library, <http://www.franconica-online.de/>.

that active learning is suitable for this real-world task. As a first step, a user indicates a rectangular crop around the map element he or she is looking for, such as  or . We use standard techniques from image processing to find a set of *candidate matches*, but the problem remains to determine which of these candidate matches are in fact semantically correct. We model this as a classification problem and use pool-based batch-mode active learning. Experiments show that the resulting human-computer interaction is efficient.

2 Related Work

Since the digitisation and analysis of historical maps is of increasing interest to libraries, several systems simplifying this complex process have been developed. Most of these systems provide convenient graphical interfaces, but still rely heavily on users to manually annotate or even georeference the input maps. See for example Fleet et al.’s *Georeferencer* [7] and the system by Simon et al. [23]. For the postprocessing of georeferenced maps, Jenny and Hurni [13] introduced a tool that is able to analyse the geometric and geodetic accuracy of historical maps and then visualise the identified distortions.

Some research has gone into image segmentation specifically for bitmap images of (historical) maps. Höhn [9] introduced a method to detect arbitrarily rotated labels in historical maps; Mello et al. [15] dealt with the similar topic of identifying text in historical maps and floor plans. These systems are rather sensitive to their parameters, requiring careful tweaking in order to perform well. In a further paper, Höhn et al. [10] specifically raise this as an area for improvement: their experiments work well, but do not necessarily generalise to a large variety of maps. The system of Mello et al. was developed for a large set of rather homogeneous maps, which means that it was merited to spend significant effort to find good parameter values. In contrast, we aim to handle diverse maps, each with relatively small user effort. We therefore specifically address finding model parameters.

There is not much research available on fully-algorithmic information retrieval specifically from historical maps. Automatic approaches exist, but only for restricted inputs—that is, developed specifically to digitise a particular corpus. For example, Leyk et al. [14] describe a method to find forest cover in a specific set of 19th century topographic maps. Arteaga [1] extracts building footprints from a set of historical maps from the New York Public Library (NYPL). The effectiveness of these approaches is in part due to the homogeneity of these relatively recent maps. The tests in this paper are performed on much older maps (16th and 18th century).

We approach the above problem using active learning (see Settles [20] for a survey). In particular, we use batch-mode learning [4, 8, 11]. Our approach is pool based, that is, we have a discrete set of items that we wish to classify and we can only query the oracle on those items. In effect, we learn a threshold [3], based on logistic regression. See Schein and Ungar [18] for a general discussion of active learning for logistic regression.

The design of our system takes into account the human factors involved in using a human as oracle. This combines aspects of human-computer interaction (HCI) and knowledge discovery, as advocated for example by Holzinger [12]. Such factors can be incorporated in the algorithms used, as in *proactive learning* [6]. For our purposes we found that standard active learning suffices.

3 Design Rationale

Our general goal is to georeference bitmap scans of historical maps. We focus on a specific subtask of this larger goal in order to get a manageable problem. This modular approach—with subgoals more modest than “understand this map”—allows for rigorous problem statements and, thereby, reproducible experiments and comparability; this is in contrast to monolithic software systems, where it can be unclear how any specific detail influences the outcome. Competing systems for a certain step can then be proposed and evaluated. Such a “separation of concerns” in systems for processing historical maps is also advocated, for example, by Shaw and Bajcsy [22] and Schöneberg et al. [19]. The latter propose a pipeline with separate tasks operating independently; our (interactive) system could serve as a module in such a system.

The task we discuss in this paper is finding pictographs and textual elements. This is an information extraction step that lifts from the unstructured level of a bitmap image to data that is combinatorial in nature: a list of locations of map elements. Finding approximate matches of an example image is a classic problem in image processing (see for example Brunelli [2] for an overview). This approach can be used for a variety of map elements, from settlement pictographs, to forests, to text labels: we find approximate repeat occurrences of an example image. However, standard techniques yield only a list of candidates along with “matching scores.” this still needs to be converted into a yes/no classification. In this paper we focus on efficiently learning a classifier in this setting.

Specifically in our application, the user provides a *template* by indicating the bounding box for an interesting map element. This could be a prototypical pictograph on the map, such as a house (🏠), a tree (🌳) or even individual characters (A, e, n). See Fig. 3 for an example: here the user wants to find all occurrences of the character ‘a’ and inputs the red rectangle in the leftmost image. The template matching algorithm comes up with—among thousands of others—the three matches indicated in the other images. The remaining problem is to decide which of these matches are in fact semantically correct.

The usefulness of recognising individual characters should not be underestimated, since standard optical character recognition (OCR) does not perform well when applied directly to an entire historical map: consider for example Fig. 1, particularly the middle image where the text is not clearly separated from the other map elements. Even in such messy maps, there are usually several characters that are particularly recognisable (which ones might depend on the handwriting). Given one typical example of a character, our method can be used to find most of the other occurrences of the character with high precision.

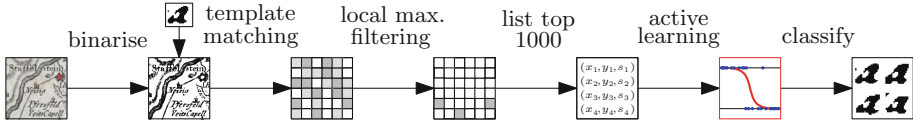


Fig. 2. Overview of the consecutive steps in our method. The input is a bitmap generated by scanning a historical map, and a template to search for. The output is a list of positive matches and their location in the image.

If we do this for a small number of different characters, a later pipeline step can cluster these results to find out where the text elements are (for example: labels). This can be used as a preprocessing step for OCR, in case the OCR algorithm would otherwise get confused by overlapping map elements or is computationally too expensive to be run on the entire map. Because of this application, we prefer our system to have a tendency to side with precision over recall: false negatives are not a disaster if we use a suitable set of characters, since it is likely that at least some character occurrences within each label will be found. This approach based on finding a small set of specific characters as preprocessing is also used by Leyk et al. [14].

In our experiments we have used a basic template matching algorithm, which we briefly sketch here. Since all our maps are effectively black and white, we first binarise to a 1-bit-per-pixel bitmap. Then we consider a sliding window, which calculates a matching *score* for every possible position, to pixel precision: when the template is shifted to a certain position, how many pixels are equal between the template and the image, and how many are different? Following standard procedure, we take the percentage of equal pixels as our matching score.³ If the score is high for a certain pixel (that is, for a certain position of the template), it is likely that a slight shift of the template still results in a good score; we therefore throw out all pixels that do not have maximal score in their 8-neighbourhood. Of the remaining pixels, we select the 1000 highest-scoring ones: this parameter is chosen generously such that all true positive matches survive this step. In this way, the template matching algorithm is used as a data reduction and projection step that takes place before the classification happens. See Fig. 2 for an overview of the different steps in this process.

This leaves the classifier. We choose to classify based on a score threshold, or equivalently: a rank threshold. A threshold that more-or-less cleanly separates the true positive matches from the true negative matches does indeed exist in our experiments: we have manually created ground truth for the templates in Table 1 and find ROC curves with area under curve of around 0.9.

Because the maps and the templates vary wildly, picking a single threshold value will not work. Some literature in fact handwaves this issue (for example [9]) by hand picking the value for their experiments. This is valid when the objective

³ Note that this basic approach is not invariant to scale and rotation. It is naturally robust against *small* variations, but some historical maps would require a more advanced template matching algorithm.

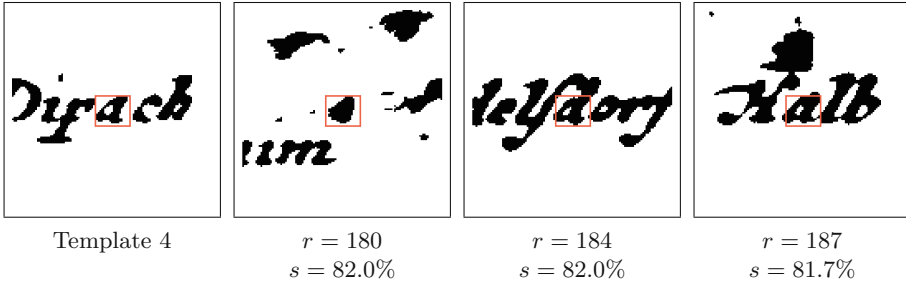


Fig. 3. Various crops from the same historical map. The red rectangle in the leftmost image indicates the crop used as template; the other three are computed candidate matches. Note that these three matches have similar rank and score, but do not all represent semantic matches of the template. In the ground truth we reject the rank-180 match (probably a hill) and accept the rank-187 match (‘a’). The ground truth of Experiment 4 accepts the rank-184 match (‘d’): see Fig. 4 for the reasoning.

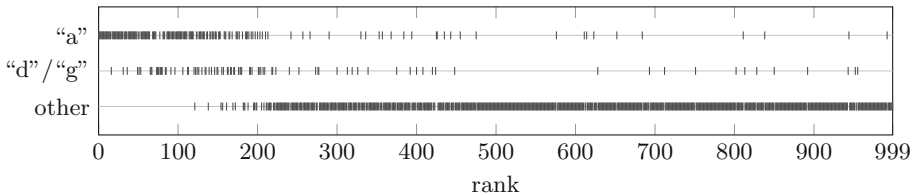


Fig. 4. Distribution of the contents of the first thousand matches for Template 4, ordered by rank. Matches containing either “a”, “d” or “g” can be separated fairly well from the remaining matches using a threshold (e.g. rank ≤ 200). In contrast, a discrimination of strictly the matches showing “a” will not have high accuracy.

is to show that a certain algorithm *can* achieve high accuracy, but does not show usefulness of the method in practice. In order to efficiently classify the potential matches given to us by the template matching algorithm, we will employ pool-based active learning with a human user as oracle.

Since a given candidate match either contains the desired element (correct) or does not contain it (incorrect), we describe it with a dichotomous variable. We then use logistic regression as a model to discriminate between correct and incorrect matches. In the experiments section we show that logistic regression is a suitable classifier when trained on complete ground truth (all labels).

Since acquiring labels is the most time-consuming step in our system—it involves a human—we use active learning. Following standard practice, we use the following batch-mode query strategy. As input it takes the list of candidate matches, ordered by rank, and a parameter k , the size of a batch. (We examine the choice of k in the experiments section.) The algorithm starts by assuming the best-scoring match is correct and the worst-scoring match is incorrect and (trivially) fits an initial model. Then, in each iteration it picks the k unlabeled

matches that are most uncertain (according to the current model) and asks the user to label this batch; the results are stored and the model is retrained. After any number of iterations, this gives the following classifier: return the user-provided label if available, and give the most likely answer according to the logistic regression model otherwise.

4 Experiments

In order to evaluate the efficacy of our method, we have implemented the proposed system and applied it to several real-world datasets. This section describes our findings.

4.1 Evaluation Settings

We implemented our method primarily in Python, using the *Scikit-learn* library⁴ for logistic regression. The template matching is implemented in C++. All experiments presented in this section have been run on a desktop PC. Neither runtime nor memory were an issue; template matching takes up to a couple of second on practical maps and batch selection occurs in realtime.

To evaluate our active learning approach, we created nine real-world data sets. These were created by analysing template matching results from actual historical maps, using various templates: the combination of a map and a template identifies a data set. For every data set, we considered the thousand highest-ranking matches and manually determined if they are correct. This gives us a ground truth containing nine times 1,000 samples; Table 1 gives an overview of these datasets.⁵ Note that for some templates we have accepted several characters, not just the exact character in the template. This improves classification performance for reasons illustrated in Fig. 4; see also Deseilligny [5]. Choosing which characters to accept for a certain template currently involves some user judgment, but the sets shown in the table seem widely applicable.










The samples in these data sets have only one feature: their score according to template-matching algorithm. These scores also imply a ranking of the samples. In each of the following experiments, there was no clear difference between using the actual scores and using the implied rank. For the rest of the paper, we report the results of using the sample’s rank as its (only) feature.

In order to assess how difficult the classification for a particular template is, and if learning is even feasible, we use ROC analysis for a threshold classifier. Figure 5 shows an area under curve of over 0.85 for all experiments, showing that the approach is feasible for a wide range of templates. As an additional measure of difficulty, we trained the logistic regression model on a full ground truth of each data set. This allows us to calculate the self information (or: surprisal) for every sample, relative to this model. Table 1 shows the sum of self information

⁴ See [17] and <http://scikit-learn.org/>.

⁵ Available at <http://www1.pub.informatik.uni-wuerzburg.de/pub/data/ds15/>.

Table 1. Data sets used in our experiments. Each line describes one data set: the name of the map, a thumbnail of the template, characters that were considered positive matches, the area under curve according to Fig. 5 and the self-information relative to the logistic regression model trained on all samples.

	Historical Map	Template	Accepted	AUC	Self-Info
1	<i>Carte Topo. D'Allemagne</i> (1787)		b, h	0.85	462.91 bit
2	<i>Franciae Orientalis</i> (1570)		a, g, d	0.90	566.95 bit
3	<i>Franciae Orientalis</i> (1570)		e	0.87	642.02 bit
4	<i>Circulus Franconicus</i> , De Wit (1706)		a, g, d	0.92	444.48 bit
5	<i>Das Franckenlandt</i> (1533)		a, g	0.87	590.50 bit
6	<i>SRI Comitatus Henneberg</i> (1743)		n, m, h	0.92	524.85 bit
7	<i>SRI Comitatus Henneberg</i> (1743)		e	0.87	524.01 bit
8	<i>Circulus Franconicus</i> , De Wit (1706)			0.88	560.29 bit
9	<i>Circulus Franconicus</i> , Seutter (1731)			0.99	146.16 bit

All maps in this table are taken from the Franconica collection (<http://www.franconica-online.de/>) of the Würzburg University Library. Identifiers: **1**: 36/A 1.16-41; **2**, **3**: 36/A 20.39; **4**, **8**: 36/A 1.17; **5**: 36/G.f.m.9-14,136; **6**, **7**: 36/A 1.13; **9**: 36/A 1.18.

over all matches of each template. This can also be regarded as a measure of the classification difficulty for the particular template: high self information hints at a larger number of outliers and/or a wider interval of rank overlap between the positive and negative samples. This interpretation is confirmed by the fact that the data sets collected on maps from the 16th century have higher self information than those on maps from the 18th century. On many of the older maps, elements indeed seem harder for humans to distinguish due to the heterogeneous style of handwriting and the suboptimal state of preservation.

We measured the classification performance of our algorithm using accuracy and F1 score. (See for example Parker [16] for definitions of these standard evaluation criteria.) Values for precision and recall of our classifier will also be discussed. Recall that for our application, precision is more important than recall: a missed character or text label might still be located later using another template, whereas false positives could potentially disturb subsequent pipeline steps (such as OCR) significantly.

4.2 Evaluation Results

Classification Performance. We have run our algorithm on the nine real-world data sets introduced above. Following Settles [21], we use *learning curves* to show the performance of our method. We use batch size $k = 3$ unless stated otherwise. A discussion of the choice of this parameter value follows later. As a baseline, we have used a random strategy, where the batch of samples to be labeled is picked uniformly at random from the pool of unlabeled samples.

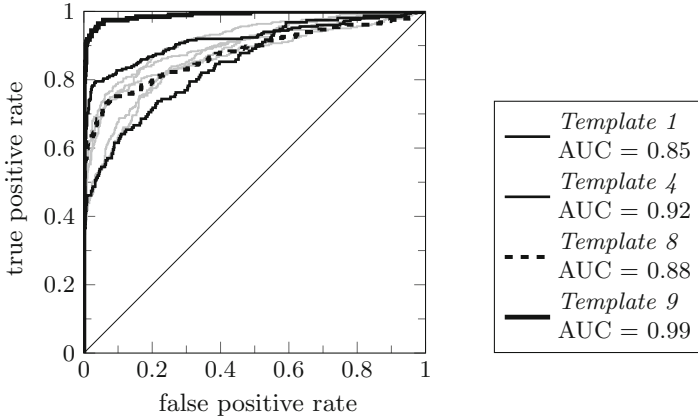


Fig. 5. ROC curves for the data sets in Table 1. Highlighted are the lowest and highest area under curve values for templates containing characters (Templates 1 and 4). Of the two templates for place markers, one shows typical performance (Template 8) and one performs exceptionally well (Template 9).

We now show that our active strategy outperforms this baseline strategy in almost every situation.

Figure 6 shows the learning curves of our active learning approach in comparison to the random strategy. The plots describe the accuracy of both classifiers against the number of iterations; the number of labeled samples is three times this number, as we set $k = 3$. For the random strategy, we performed 100 runs and show mean, 10th, and 90th order statistic of the achieved accuracy. As we can see in the figure, the accuracy of the active learning strategy dominates the accuracy of the random strategy at almost every iteration. Only in the very beginning (number of iterations below approximately 15), this is not consistently true. However, the active learning approach is near the 90th percentile performance of the random strategy in these situations as well.

Now we look at additional performance measures. The results in this experiment refer to Template 6, as a typical example. Figure 7 shows the performance of the active learning approach in comparison to three runs of the random sampling strategy. Note that after 15 iterations, the active learning classifier dominates the three random classifiers in accuracy, precision and F1 score. The random approach does better only in terms of recall, which we find acceptable, as discussed before. The same observations hold for a larger number of random runs and for the remaining data sets; plots are omitted for space.

It can additionally be noted that, in contrast to the random approach, all four scores increase monotonically after the first few iterations when using the active learning method. Thus, when adding additional labels, the classifier’s performance is highly likely to improve. This property is especially valuable for the design of proper user interaction when using active learning: from the users’

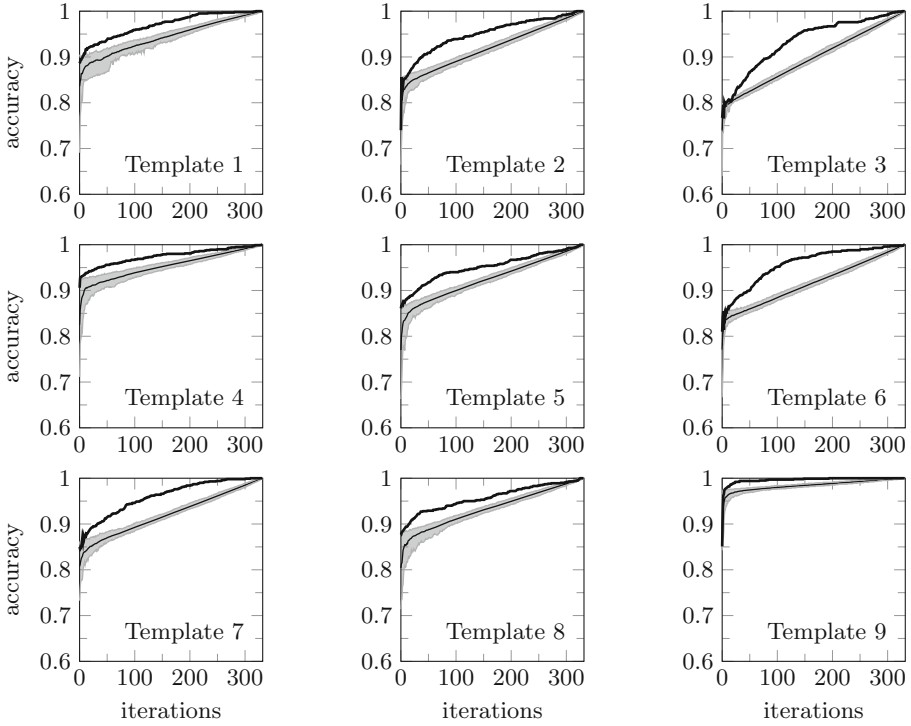


Fig. 6. Learning curves comparing the performance of our active learning strategy ($k = 3$) to the random baseline strategy. The bold black line indicates the accuracy of our method over the iterations. The thin black line shows the mean accuracy of 100 runs of the random strategy; the grey area indicates 10th to 90th percentile.

point of view, it is hard to accept that additional effort in labeling leads to a decrease in quality.

In the next experiment, we consider the self information of the samples that our system selects, in comparison to those chosen by the random baseline strategy. We calculate the self information as before (see Evaluation Settings). For almost any number of iterations, the total self information in the samples from the active learning strategy is considerably higher than in those from the random baseline strategy. Figure 8 illustrates this for four templates; the same holds for the remaining five data sets. This behaviour of the active learning strategy is desirable, because higher self information means that the labeled samples were *indeed* hard to classify for the logistic regression model and therefore having them labeled by the user is valuable. In contrast, the random strategy presents a substantial number of samples whose labels are comparatively clear (for example because they have a very high rank), thereby wasting the user’s time.

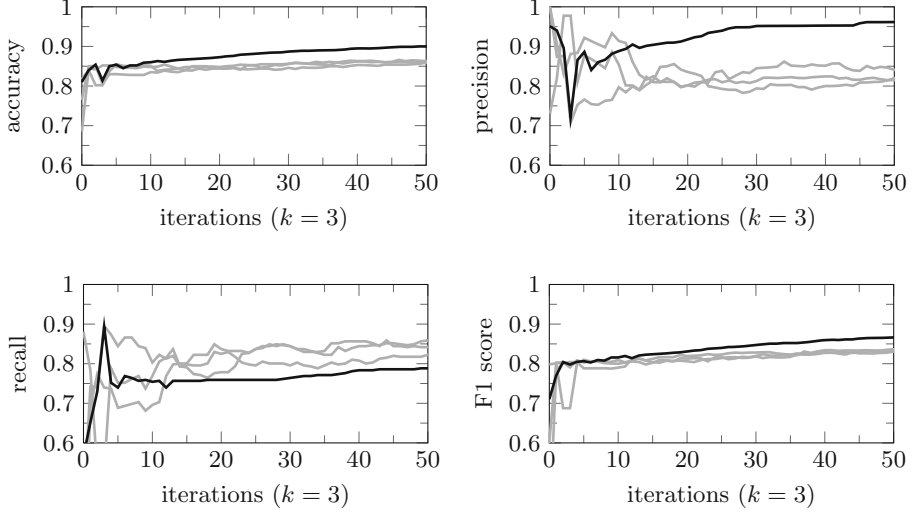


Fig. 7. Statistics for our active learning strategy (black) and three runs of the random baseline strategy (grey) on Template 6. Note that after 15 iterations, this strategy outperforms the random baseline strategy in all measures except recall.

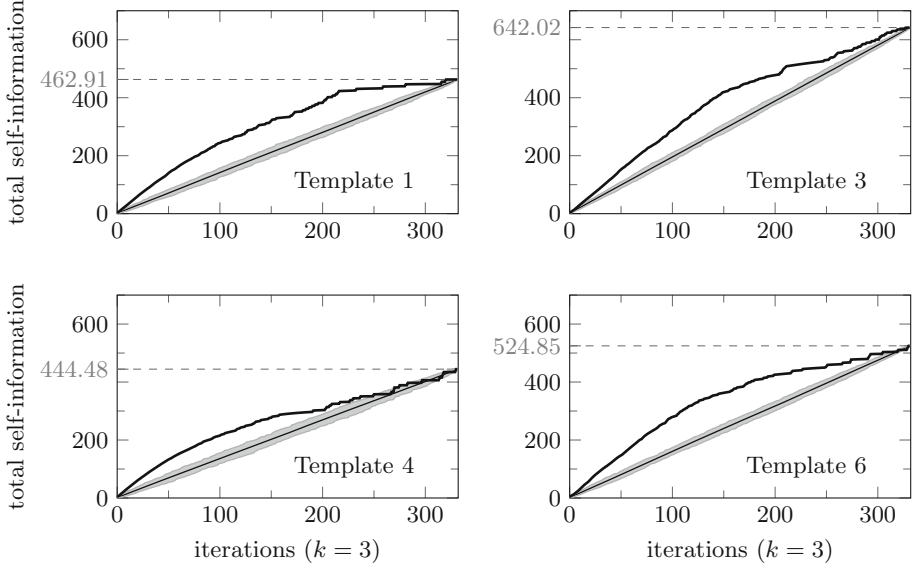


Fig. 8. Self-information of all samples that have been labeled up to a given iteration. The labels picked by the active learning strategy (bold black) are considerably more informative than those selected by the random baseline strategy (black: mean, grey area: 10th to 90th percentile). Note that in the end, each strategy has labeled all samples and achieves the self information of the ground truth as listed in Table 1.

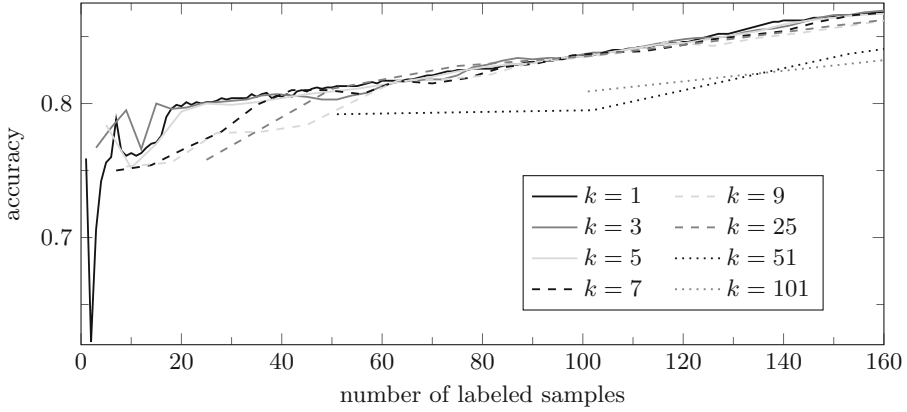


Fig. 9. Accuracy of our active learning strategy using different batch sizes k on Template 3. For values of k between 3 and 7, accuracy is acceptable from the start and increases for increasing number of samples. Exceedingly large values ($k \geq 25$) result in inferior performance for the first few iterations and these represent significant user effort due to the batch size.

Runtime. In our decidedly unoptimised implementation, it takes a total of approximately one second of runtime to calculate 100 batches of size $k = 3$. As this represents 100 batches of user interaction, the system is clearly suitable for realtime applications.

Choice of Parameters. Our system depends on the batch size k . We have run a set of experiments to evaluate the influence of k on our system’s classification performance. Figure 9 shows that the performance of our system does not depend very strongly on the choice of k , as long as no extreme values are chosen. Based on this data set, we might recommend values between 3 and 7. This conclusion holds for the remaining templates (plots omitted for lack of space).

When choosing the parameter k , human factors should also be taken into account. The time taken to decide if a displayed candidate match is correct (that is, to label a sample) varies with the batch size. Since selecting and delivering a new set of samples to the user requires a perceptible amount of time (both technologically and cognitively), a larger batch size may cause less user disturbance. For this reason—and aesthetic reasons—we currently use $k = 9$ in our web-based prototype implementation of the user interface.

5 Prototype of a User Interface

In addition to the experimental setup described above, we have implemented a prototype user interface for our system. This allows us to assess not only the abstract, but also the practical suitability of our approach. Figure 10 shows two screenshots of our implementation. The interface on the left allows users

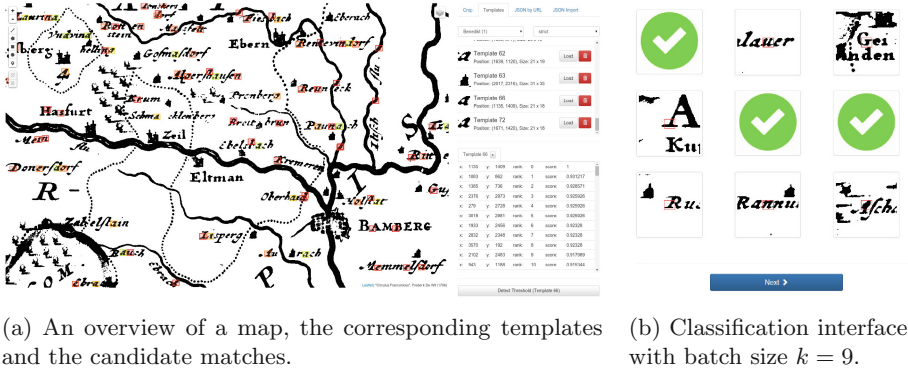


Fig. 10. Screenshots showing two user interfaces from our web-based prototype. Note that (a) is intended to be used on large screens, while (b) can be used conveniently on smartphones as well.

to browse a historical map, crop templates and start the template matching process. With the interface on the right, users can classify samples selected by the active learning system (in the screenshot $k = 9$). By clicking or touching any of the nine tiles, the tile turns around and shows a green check mark to indicate that the sample was classified as positive. Once the user is finished inspecting the nine samples, he or she presses the “Next” button. The samples that remain unchecked will be considered negative and a new batch of samples chosen by the active learning algorithm will be presented. Our implementation of the user interface is web-based (using HTML5 and JavaScript), so it can be used seamlessly on any device that runs a modern browser. In particular, the classification interface can be conveniently used on smartphones, which enables crowdsourcing of this task. (See also Arteaga [1].)

Using our prototype, it takes a user with some experience approximately 25 seconds to do 4 iterations (that is, to classify 36 samples, since $k = 9$). This includes the runtime of our active learning algorithm and client-server overhead. According to our experimental results in the preceding section, the stated number of labels is already enough to achieve good classification results for a typical template. Projecting these numbers, our approach allows the effective classification of 10 templates within 5 min, assuming the templates have been selected beforehand. In contrast, even with significant experience it takes about 10 to 15 min to generate the full ground truth for a single template. This leaves quite some time to select the templates and still achieve a factor-10 improvement in template throughput. (Recall that the user is probably looking for many templates on the same map.) This shows that our system, and the proposed user interaction, is well-suited for this application.

6 Conclusion

In this paper, we have tackled a real-world problem from a knowledge-discovery perspective: the extraction of information from historical maps. We have focused on the detection of occurrences of certain elements in bitmap images, and introduce a practical approach that solves this problem. Our proposed system uses template matching for feature extraction from the image, and batch-mode active learning to detect appropriate thresholds. Particularly this active-learning step addresses an open problem in the literature on metadata extraction from historical maps. We implemented this approach and experimentally demonstrate that it performs well on real data sets from practice. In combination with the user interface we propose, our system is able to save users a significant amount of time when georeferencing historical maps. Directions for future work include the following.

In a practical setting, our system clearly extends to other (historical) documents besides maps. Early experimentation shows, for instance, that the system also works well for locating specific glyphs in medieval manuscripts. Our prototype is currently being integrated into the existing workflow at Würzburg University Library, which will enable user studies on a proper scale.

On a more abstract level, our active-learning approach with human-computer interaction is not limited specifically to historical documents and template matching. We expect that many other computer-vision methods that depend sensitively on parameter selection can benefit from this strategy.

Acknowledgments. We thank Wouter Duivesteijn for fruitful discussion and helpful comments. We thank Hans-Günter Schmidt of the Würzburg University Library for providing real data and practical use cases.

References

1. Arteaga, M.G.: Historical map polygon and feature extractor. In: Proceedings of the 1st ACM SIGSPATIAL International Workshop on MapInteraction, pp. 66–71 (2013)
2. Brunelli, R.: Template Matching Techniques in Computer Vision: Theory and Practice. Wiley, New York (2009)
3. Bryan, B., Nichol, R.C., Genovese, C.R., Schneider, J., Miller, C.J., Wasserman, L.: Active learning for identifying function threshold boundaries. *Adv. Neural Inf. Process. Syst.* **18**, 163–170 (2006)
4. Chen, Y., Krause, A.: Near-optimal batch mode active learning and adaptive submodular optimization. In: Proceedings of the 30th International Conference on Machine Learning, pp. 160–168 (2013)
5. Deseilligny, M.P., Le Men, H., Stamon, G.: Character string recognition on maps, a rotation-invariant recognition method. *Pattern Recogn. Lett.* **16**(12), 1297–1310 (1995)
6. Donmez, P., Carbonell, J.G.: Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 619–628 (2008)

7. Fleet, C., Kowal, K.C., Pridal, P.: Georeferencer: crowdsourced georeferencing for map library collections. *D-Lib Mag.* **18**(11/12) (2012)
8. Guo, Y., Schuurmans, D.: Discriminative batch mode active learning. In: *Advances in Neural Information Processing Systems 20, Proceedings of the 21st Annual Conference on Neural Information Processing Systems*, pp. 593–600 (2007)
9. Höhn, W.: Detecting arbitrarily oriented text labels in early maps. In: Sanches, J.M., Micó, L., Cardoso, J.S. (eds.) *IbPRIA 2013. LNCS*, vol. 7887, pp. 424–432. Springer, Heidelberg (2013)
10. Höhn, W., Schmidt, H.G., Schöneberg, H.: Semiautomatic recognition and georeferencing of places in early maps. In: *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 335–338 (2013)
11. Hoi, S., Jin, R., Zhu, J., Lyu, M.: Batch mode active learning and its application to medical image classification. In: *Proceedings of the 23rd International Conference on Machine Learning*, pp. 417–424 (2006)
12. Holzinger, A.: Human-computer interaction and knowledge discovery (HCI-KDD): what is the benefit of bringing those two fields to work together? In: Cuzzocrea, A., Kittl, C., Simos, D.E., Weippl, E., Xu, L. (eds.) *CD-ARES 2013. LNCS*, vol. 8127, pp. 319–328. Springer, Heidelberg (2013)
13. Jenny, B., Hurni, L.: Cultural heritage: studying cartographic heritage: analysis and visualization of geometric distortions. *Comput. Graph.* **35**(2), 402–411 (2011)
14. Leyk, S., Boesch, R., Weibel, R.: Saliency and semantic processing: extracting forest cover from historical topographic maps. *Pattern Recogn.* **39**(5), 953–968 (2006)
15. Mello, C.A.B., Costa, D.C., dos Santos, T.J.: Automatic image segmentation of old topographic maps and floor plans. In: *Proceedings of the 2012 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 132–137 (2012)
16. Parker, C.: An analysis of performance measures for binary classifiers. In: *Proceedings of the 11th International Conference on Data Mining*, pp. 517–526 (2011)
17. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
18. Schein, A.I., Ungar, L.H.: Active learning for logistic regression: an evaluation. *Mach. Learn.* **68**(3), 235–265 (2007)
19. Schöneberg, H., Schmidt, H.G., Höhn, W.: A scalable, distributed and dynamic workflow system for digitization processes. In: *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 359–362 (2013)
20. Settles, B.: Active learning literature survey. Computer Sciences Technical report 1648, University of Wisconsin-Madison (2010)
21. Settles, B.: Active Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan and Claypool Publishers, San Rafael (2012)
22. Shaw, T., Bajcsy, P.: Automation of digital historical map analyses. In: *Proceedings of the IS&T/SPIE Electronic Imaging 2011*, vol. 7869 (2011)
23. Simon, R., Haslhofer, B., Robitza, W., Momeni, E.: Semantically augmented annotations in digitized map collections. In: *Proceedings of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries*, pp. 199–202 (2011)