Location of Simple Graphemes in Mediaeval Manuscripts based on Mask R-CNN*

Simone Marinai[1], Gabriella Pomaro[2], Claudia Raffaelli[1], and Francesco Scandiffio[1]

[1] University of Florence, Department of Information Engineering
Via di Santa Marta 3, 50139 Florence, Italy
simone.marinai@unifi.it
{claudia.raffaelli,francesco.scandiffio}@stud.unifi.it
[2] Società Internazionale per lo Studio del Medioevo Latino
Via Montebello, 7, 50123 Florence, Italy
gabriella.pomaro@sismelfirenze.it

Abstract. In this paper we describe a system for the location of simple graphemes in mediaeval manuscripts based on the Mask R-CNN convolutional neural network. This is the first step towards the ambitious goal of providing palaeographers with a powerful tool with which to speed up and refine the delicate process of dating and determining the origin of manuscripts. In order to train the network, a new dataset composed of 49 pages of Latin Middle Ages manuscripts has been built. Experimental results demonstrate that using the Mask R-CNN network, along with a proper configuration of parameters, leads to good overall outcomes of classification.

Keywords: Character recognition · Grapheme classification and location · Mask R-CNN · Deep learning · Mediaeval Manuscripts · Paleography

1 Introduction

The analysis of manuscripts, in particular their dating and localization, represents the principal way to reconstruct our history before the invention of movable type printing. Unfortunately, for the large majority of manuscripts there is no reliable information about their origin and provenance. As a matter of fact, only after the late 14th Century we have significant quantities of items with associated dating information in libraries and archives. For this reason, palaeographers use a variety of methods to determine the age of a manuscript, but they can usually only provide an approximate period of time about its origin. Among the methodologies used by palaeographers we can list the study of the material, the

* Research supported by Fondazione Cassa di Risparmio di Firenze

ink used, and of course the analysis of the language. In the case of mediaeval manuscripts, researchers have an additional element from which to draw important information to carry out their dating task: the analysis of the shape of graphemes and the features that make them up. Indeed, different ways of writing the same grapheme have spread among the amanuenses, in a way similar to what happens to us today with cursive and capital letters. Since these graphic signs have changed several times and spread slowly over the centuries, they are a trace of how writing has changed over time. By closely observing the changes in lettering, palaeographers can provide a basic time frame for when the document was written. However, some writing styles lasted for a so long time or were so widespread that they could not provide any useful information for dating. For these reasons scholars are interested in examining for each manuscript the copyist's graphic choices and also the presence or absence of significant graphic variants. This process is very long and prone to human errors, such as wrong reading or missing an occurrence of the searched sign. When manuscripts consist of a large number of pages i.e. hundreds or thousands of pages, it is very difficult to completely and carefully inspect them because it would be an extremely time consuming task. Usually, only a small amount of sample pages is analysed in order to extract the required information. This kind of approach can easily lead to incorrect dating results due to the fact that copying a manuscript took years of time during which even the same amanuensis could change its writing style several times.

For these reasons, a system capable of extracting information about the presence of certain palaeographic letter variants within a collection of documents can be of particular use to palaeographers. In this paper we describe the first version of a grapheme-detection system based on the *Mask R-CNN* deep neural network in order to identify and count the occurrences of a specific subset of graphemes in manuscripts from the Latin Middle Ages. This work is part of the project "*Mediaeval manuscripts of Tuscany (XIII-XIV centuries): design and development of software for dating and determination of origin*" financed by SISMEL (Società Internazionale per lo Studio del Medioevo Latino), DINFO (Dipartimento di Ingegneria dell'Informazione of the University of Florence), and Fondazione Cassa di Risparmio Firenze.

The remainder of the paper is structured as follows. In Section 2 we present related work about character recognition and the Mask R-CNN framework. In Section 3 we describe the building process of the dataset employed in this project. In Section 4 we present the approaches used to train the network. Finally, Section 5 contains the obtained results and in Section 6 we draw the conclusions.

2 Related Work

In this section we discuss related work concerning the detection of characters in manuscript images and we provide a summary of the main object detection approach considered in our work.

2.1 Character detection in manuscripts

Sheng et al. [1] claim that since automatic document reading does not always allow to fully understand documents, additional techniques that go beyond the mere use of OCR systems are needed. In particular, with respect to manuscripts, locating the geographic origin or identifying the writer may be also relevant tasks. For this reason the authors have decided to develop a particular set of features that allows to map the pixels composing the characters in an high-dimensional space, capturing in this way specific information about the characters. The proposed features can be used separately or jointly and are based on the principle of *Joint Feature Distribution* (JFD). The goal is to answer four questions in the field of palaeography about who produced a certain document, which document, when and where. The proposed features are divided between Textural based features and Grapheme based features. The first ones consider the manuscripts as textual images, extracting statistical information from the text blocks on the entire image. They capture the curvature and skew characteristic of different writing styles and typically do not require line or character segmentation. As for the grapheme-based features, these allow to capture the statistical distribution of the single character already segmented starting from documents. These are based on the principle of the JFD to concatenate spatial information to obtain a larger structure that is faithful to the traced sign. Among the features that best allow to date a document, in their work the authors highlight CoHinge and QuadHinge [2], which fall into the category of textural-based features, as well as the Junction feature (Junclets) [3] with regard to grapheme-based features. Wick et al. in [4] propose a method for the automatic transcription of lyrics in mediaeval music manuscripts. The work is based on the open-source OCR engine Calamari [5]. The predictions are made on previously segmented lines of the original manuscript page using an available pre-trained model or with a custom model. In [6], Wahlberg presents a method for line segmentation, along with a set of features that can be used for text recognition, writer identification and production dates.

2.2 Mask R-CNN

Artifical Neural Networks ,and in particular deep learning architectures, have bee widely used in to process historical documents [7]. Among other approaches, Mask R-CNN is one state of the art model for instance segmentation and object detection, developed by Facebook AI Research group [8]. Mask R-CNN extends Faster R-CNN (already used to locate words in early printed documents [9]) making use of an extra mask head and is composed by a standard convolutional network for image classification and, on the top of that, an additional fully convolutional network for semantic segmentation at pixel level on the proposed regions. The network undergoes through two main stages. The first one is responsible for generating, on the input image, a set of proposals i.e. regions where there might be an object. The second one is related to the output produced by the network and is designed for classifying the proposal suggested at

(a) S Dritta. (b) S Tonda. (c) S Documentaria. (d) D Dritta. (e) D Tonda. (f) K.

(g) Z. (h) Z3. (i) C Cedigliata. (j) Et in legatura. (k) Et tachigrafica. (l) T assibilata.

Fig. 1: Graphemes of interest to be detected.

the previous stage, in order to allow bounding boxes and masks generation. At the end of the process, the result is a bounding box that encloses the recognised object and a pixel mask placed on it. The detection branch, that is the branch for classification and bounding box, runs in parallel with a branch used for predicting segmentation masks, allowing a decoupling of the two tasks. For more details refer to [8].

3 Building the dataset

It is important to remark that a program aimed at identifying simple graphemes should not be designed taking into account one specific period for the manuscript production or one particular region. Rather, it should focus on the peculiarities of the graphic material to be examined, in which the simple grapheme has a specific meaning. Even if the approach we propose is not designed to deal with cursive writings and would not work for historical periods in which the syntagm is more important than the paradigm, there are whole centuries whose manuscripts can be suitably analysed and on which we focus our research: they are manuscripts between the XIth and the XIVth centuries and also documents from the XVth century.

The manuscripts used in this study are carefully selected from the large Codex archive that has been built by SISMEL in the last twenty years by cataloguing mediaeval manuscripts from Tuscany in the Codex Project[3]. In particular, the data used are based on the ample collection of scanned works accurately linked to the codicological descriptions. Taking into account the period of time previously mentioned, we believe that it is important to identify - and compute the distribution in the manuscripts - of the following graphemes: three variants of the letter *s* and two of the letter *d*; *ligature et*; *tachygraphic et*; *k*; three variants of *z* (including the *ç*). Given the peculiarities of the dataset, we are also considering to include the graphic sign *ti assibilata*: this is a ligature of *t* and *i*

[3] https://www.sismelfirenze.it/index.php/biblioteca-digitale/codex

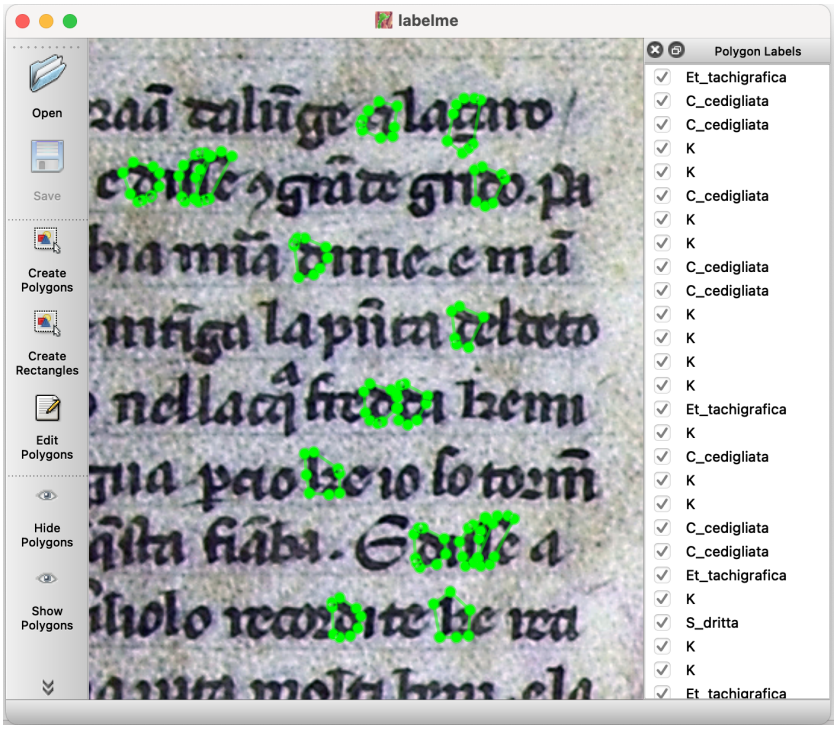


Fig. 2: Example of an annotated image viewed from the Interactive LabelMe interface.

that is however an isolated symbol. Currently, the data collection process is still in progress, so this last grapheme will be introduced in a later version of the dataset. The set of graphemes of interest is shown in Figure 1.

In order to locate and recognize the graphemes of the subset above, it was necessary to build an appropriate dataset[4] with Latin Middle Ages characters from the period XI-XIV *ineunte*. The examples were gathered from manuscripts of the accessible digital databases, giving preference to those originated from Tuscany. To achieve good training of the neural network, only documents without excessive signs of wear such as burns, rubs and tears were selected. However, it should be noticed that online libraries of ancient documents usually expose only low-quality images to the public. The difficulty of collecting good quality pages suitable for our purpose inevitably influenced the quantity of images that make up the dataset. The dataset consists of 49 pages of which only 11 are completely labelled. Other documents have already been identified but have not been validated; we plan to include the future release of the dataset a much larger number of pages and examples.

The occurrences of the graphemes of interest have been manually annotated through the Interactive LabelMe program [10, 11] (Figure 2) which outputs a

[4] The manuscripts that make up the dataset are listed in the Acknowledgements section and are available upon request by sending an email to the authors of this paper.

Table 1: Number of labelled graphemes in each split of the dataset grouped by class.

Class	Train	Validation	Test	Total
S dritta	1009	441	533	1983
D tonda	933	329	358	1620
S tonda	367	70	48	485
Et tachigrafica	279	39	26	344
D dritta	192	21	47	260
K	61	27	67	155
Et in legatura	106	21	18	145
C cedigliata	66	17	20	103
Z3	40	25	20	85
S documentaria	44	15	5	64
Z	16	2	2	20

JSON file containing polygonal segmentations of the graphemes. The JSON files have been converted into the COCO (Common Objects in Context) format for ease of use with Mask R-CNN. Since characters are usually drawn very close to each other, some annotations contain not only the grapheme of interest but also small portions of adjacent signs. This implies that some of the segmentations have little noise which was nevertheless deemed acceptable.

Some of the images in the dataset have a very high resolution. This has a positive effect on learning but is also a challenging element for the amount of GPU memory required for training. After investigating various cutting methods, we decided to cut each image into four blocks of the same size. Since manuscripts do not have a predetermined page structure (in some documents the text is a continuum without separation into columns while in others it surrounds figures that can be placed anywhere in the page) all images are processed in the same way. Annotations along the cut lines are discarded.

Since the dataset is to be used for a very specific task, we decided to rely only on the content of carefully selected manuscripts, avoiding the use of data augmentation techniques. As a consequence of this choice, considering also that the Latin language has some graphemes much more frequent than others, the number of annotated characters is strongly unbalanced in favour of some common classes and is almost totally non-existent for other rare - but still important - palaeographic letter variants (see Table 1). For instance, *S dritta* and *D tonda* are the most frequent classes, making up 70% of the dataset. Such an unbalanced set of data can create some learning issues. It is indeed highly probable that, with this configuration, the network will learn well the most common classes, and not so well the rarest ones. Finally, the dataset has been divided into train, validation and test sets. Since a complete and reliable ground truth is critical to a proper performance evaluation, the 11 fully annotated pages have been divided

between validation (5) and test (6). The remaining 38 documents were used for training.

4 Model Identification

In this section we present the experiments conducted to adjust the hyper-parameters to be used in the network training, discussing the effects produced by their variations and explaining how they led us towards the final model.

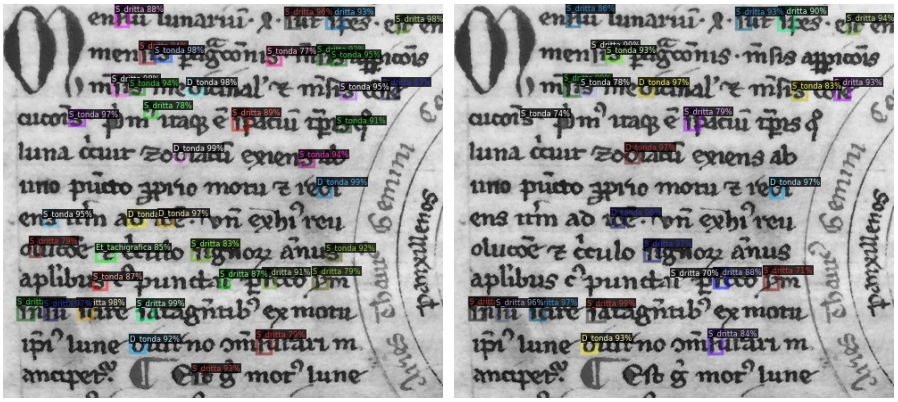
As previously discussed, this work is based on Mask R-CNN which is one state of the art convolutional network used for object detection and image segmentation. In particular, we selected the Detectron2 implementation developed by Facebook AI Research Group [12]. To configure the network we used the *fine-tuning paradigm* which consists of initialising the weights with a pre-trained model. This approach is useful when training the network from scratch is made difficult by limited amount of data available, as pointed out by a variety of scientific publications [13–18]. The chosen model is a ResNet50 with a FPN backbone trained for 37 epochs on the COCO dataset. After selecting the model, it was necessary to refine the training parameters, paying particular attention to learning rate, the number of iterations and the batch size.

In order to identify the optimal learning rate we have carried out 69 independent trainings composed of 500 iterations each, assigning to the i-th training the lr_i computed as $lr_i = 0.0001 \cdot i$. The aim of the experiment was to compute the loss calculated on the train at the end of the 500 iterations, selecting the lr with the highest variation towards the minimum value of loss. We observed that from iteration $i = 40$ the loss increases, diverging at iteration 69. With low values of lr (magnitude of 1e-3) we saw a good reduction, but the loss was still high. We have therefore decided to keep the learning rate of 3.5e-3 which combines an overall reduction with the minimum global value of loss.

The learning rate schedule and the total number of iterations have been identified through an experimental *trial and error* approach. All other configuration parameters being equal, we evaluated different scheduling policies and max number of iterations by comparing the Precision, Recall and $F1$ measures calculated on the validation set. Regarding the policy of variation, the best results have been achieved by training the network with *fixed learning rate* of 3.5e-3 for a total of 1150 iterations, obtaining the following values: Precision = 0.869, Recall = 0.551, $F1 = 0.675$. Similar but slightly worse results were obtained by training with lr fixed at 3.5e-3 for the first 800 iterations, then proceeding with a lr of 5e−4 for other 400 iterations.

Concerning the number of iterations with fixed learning rate, shorter training provides a precision in the range of ± 0.03 from the one of the selected model. As an opposite case, training for more than 1150 iterations increases precision by 5 percentage points, but at the same time negatively affects recall, bringing it down to below 0.20. This trend is confirmed by the comparison of inference boxes (Figure 3). An excessive number of iterations increases the confidence and reduces false positives but at the same time makes the model no longer able to

detect graphemes that were previously retrieved. This behaviour is attributable



(a) Test with iteration parameter set to 1150. (b) Test with iteration parameter set to 1250.

Fig. 3: Prediction results on validation produced with different number of training iterations. When iterations exceed 1150, box predictions confidence increases but many instances previously detected detected are no longer recognised.

to the fact that graphemes of the same class are written in slightly different ways over the dataset, depending on the style of the writer. For instance, some graphic signs can be traced in a more slanted way, can be larger than others and more generally present very personal characteristics related to the hand of the writer. All of this not to mention the fact that some graphemes are more likely to be drawn close to each other, making it even more difficult to distinguish them. Keeping all of this in mind, it is not surprising that an excessively high number of iterations brings the network to overfit on the style of the grapheme used in the train set, making it difficult to locate the others.

The last hyperparameter to be tuned is the *batch size*, i.e. the number of training samples that are analysed by the network before performing a weight update. The developers of Mask R-CNN adopted an *"image-centric training"* with the consequence that the batch size corresponds to the number of images analysed by the GPUs for each weight update. We choose a batch size of 16, thus computing 4 documents at a time since each document is divided into four images.

5 Result and Analysis

The fine-tuning paradigm discussed in the previous section had been actually used even at an earlier stage of this work, when the available dataset was approximately only 25% of the current size. Considering the dataset expansion work carried out over the last few months, we questioned the usefulness of initialising the network with a pre-trained model, thus investigating alternative methods of initialisation. For this reason, inspired by [19, 20], we have made experiments

Table 2: Number of annotations of the two baseline trainings grouped by class.

Class	Initial Train set	Current Train set
S dritta	530	1009
D tonda	395	933
S tonda	78	367
Et tachigrafica	85	279
D dritta	28	192
Et in legatura	3	106
C cedigliata	32	66
K	22	61
S documentaria	4	44
Z3	6	40
Z	1	16
Totals	**1184**	**3113**

on training from scratch, Furthermore, in this section we analyse how a highly unbalanced dataset can influence network metrics to appear better than they actually are.

Table 2 summarises the training examples of the two reference datasets grouped by classes. It is easy to observe that due to the intrinsic rarity of some graphemes, the example instances are not properly balanced. Moreover, more than half of the classes in the initial dataset have fewer than 35 annotations, an insignificant number compared to the great variety of sign executions that can be found even within a single manuscript.

5.1 Models comparison

Throughout the analysis of the results we decided to prefer a higher recall even at the expense of precision, provided that the value of the latter was at least 80%. The reason for this choice is related to the final objective of the research project. If the automatic grapheme identification and localisation system had a high recall and low precision it would provide many wrong results, leading the user to not trust the system and requiring to manually check a large amount of retrieved data. This would surely discourage the use of the software and therefore must be avoided. Even the opposite situation - high precision, low recall - would be counterproductive because it would provide data that is unrepresentative and not able to satisfy the research, leading the palaeographer to search manually for important but undetected graphemes. However, we would like to point out that usually the analysis of the writing is manually carried out and that due to the complexity and heaviness of the task it is usually done only on a very limited selection of pages. For this reason obtaining even only a third of the instances of a manuscript would be a considerable improvement.

Table 3: Comparison of the key metrics calculated on the validation set and test set on different network configurations.

	Validation			Test		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Network 0H	0.857	0.452	0.592	0.841	0.446	0.583
Network 1H	0.869	0.551	0.675	0.869	0.490	0.624
Network 0L	0.903	0.510	0.651	0.854	0.528	0.653
Network 1L	0.861	0.564	0.682	0.839	0.610	0.706

Considering the two datasets and the two techniques for initialising the weights, we can identify the following scenarios to which we associate codes for the sake of brevity: training from scratch on the initial dataset (0L); training from scratch on the updated dataset (0H); pre-trained weights and small dataset (1L); pre-trained weights and updated dataset (1H). By applying the approach discussed in Section 4 we obtained the following 4 models:

- **0L** is trained for 1400 iterations with fixed learning rate at 0.0035
- **1L** is trained for 1400 iterations with fixed learning rate at 0.0035
- **0H** is trained for 800 iterations with fixed learning rate at 0.0035
- **1H** is trained for 1150 iterations with fixed learning rate at 0.0035

In the first analysis we make pairwise comparisons between the models grouping by weight initialisation method and structure of the training set (Table 3). Comparing 0H and 1H it is evident that the network initialized with pre-calculated weights has better recall and precision values than its untrained counterpart. This is justified by the fact that although the dataset is better supplied with examples, these are not sufficient to allow a good training of the network without the support of a basic model.

Comparing the models 0L and 1L on the validation set it emerges that, given an equal length of training, initialising with pre-trained weights brings a benefit in terms of recall (+0.054) at the expense of precision which instead decreases by 0.042 points. From the analysis of the evaluation metrics, as the number of iterations varies 0L shows a trend that grows smoothly on both precision and recall, going into overfitting after iteration 1400. The 1L metrics, on the other hand, are more abrupt, oscillating several times before reaching the values previously reported. These behaviours are in line with what was discussed in Section 4. The trend of the validation set is confirmed by the results obtained on the test set.

When comparing the results on the test of all the networks, 1L and 0L models obtain the best scores, ranking first and second respectively for F1 measure. From these values it may seem that the models trained on the intial dataset are better than those on the updated version. This would mean that the update of the dataset made things worse, an unlikely behaviour if we compare the composition of the two datasets. Although the problem of imbalance is still present, albeit in a slightly reduced form: more than half of the classes exceeded the 100-annotation

Table 4: Comparison of the precision and recall metrics obtained on the test set and grouped by classes for both network configurations 1L and 1H.

	Network 1L		**Network 1H**		
Class	**Precision**	**Recall**	**Precision**	**Recall**	**Test set**
S dritta	0.809	0.835	0.869	0.650	533
D tonda	0.919	0.664	0.952	0.332	358
S tonda	1.0	0.042	0.823	0.292	67
Et tachigrafica	0.611	0.423	0.733	0.423	48
D dritta	0.0	0.0	1.0	0.149	47
K	0.0	0.0	0.704	0.567	26
Et in legatura	0.0	0.0	0.882	0.833	20
C cedigliata	0.0	0.0	0.857	0.3	20
Z3	0.0	0.0	0.0	0.0	18
S documentaria	0.0	0.0	0.0	0.0	5
Z	0.0	0.0	0.0	0.0	2

threshold, becoming more significant in the training phase. Since this can only be a positive fact, a deeper analysis is necessary.

First of all, we note that in this context it is much more useful and accurate to assess precision and recall separately for each class, as in Table 4. This method of analysis is necessary to take into account the different probabilities of occurrence of Latin graphemes, element that inevitably affects the number of examples in the dataset. However, every grapheme in the set of interest has relevance and this is why it would be unacceptable to produce good results on only a subset of the selected classes.

From the results grouped by class (Table 4) it is clear that 1L cannot provide information on more than half of the characters and is therefore an unsatisfactory model. This result can be explained by looking at the structure of the initial training set: the 1L model has been able to specialise exclusively on the recognition of the first four characters with the largest number of examples.
Comparing 1H and 1L we can say that having a lower recall on the most common classes and a higher recall for all the others is a good indicator that the dataset update has succeeded in preventing the network from specialising on a subset of characters, bringing us closer to the final goal. Certainly further work needs to be done to increase the number of examples relating to the graphemes *Z3, S documentary, Z*, which are currently not recognised by the network because they are last in terms of number of annotations.

Summarizing, the information on the performance of the models contained in Table 3 expresses a value that may seem absolute but that in reality is closely related to the structure of the training set used. It is therefore incorrect to use the values in Table 3 to compare L models with H models and say that L models are preferable to H models because they achieve better metrics. Instead, it is

correct to say that 1L and 1H are better than 0L and 0H respectively, i.e. that initialization with pretrained weights produced better results than initialization from scratch.

6 Conclusions

The content of this paper is part of a larger project which aims to provide palaeographers with a software able to classify and locate simple graphemes within mediaeval manuscripts. The major benefit of an automatic detection will be to replace the time-consuming process of manual analysis with a quick and easy way of obtaining information about simple graphemes contained within entire document collections. The core of the system is a Mask R-CNN network trained to recognise a specific subset of graphemes. The training phase was carried out on a new dataset built by manually labelling images of manuscript from the period XI to XIV. The results discussed in Section 5 show that among the proposed models the best results are achieved by the pre-trained network. This is justified by the fact that the amount of data available needs to be further increased and better balanced before dropping the use of pre-trained weights. Training from scratch provided satisfactory results, although slightly worse than its pre-trained counterpart.

The first future development to be carried out concerns the improvement of the dataset. Expanding the dataset by adding examples for rarer classes would help in increasing the recall for those classes. In order to enhance the quality of the training dataset another technique that could be applied is the one of data augmentation in favour of those classes with fewer examples.

Acknowledgements

This work is partially supported by the *Fondazione Cassa di Risparmio di Firenze* that funded the project *I manoscritti medievali della Toscana (sec. XIII-XIV): progettazione e sviluppo di un software per la datazione e la determinazione di origine* granted to SISMEL.
We would like to thank the following libraries for providing us the documents:

- Barcellona, Biblioteca de Catalunya: 639 f. non precisabile
- Bologna, Biblioteca Universitaria: 1746 ff. 7, 8, 10, 30, 42, 52
- Berlin, Staatsbibliothek zu Berlin: Rehdiger 227 f. 10r; Phillips 1716 12v, f. 39r
- Cologny, Fondation Martin: Bodmer 30 f. 12v
- Firenze, Biblioteca della Fondazione E. Franceschini: ms. 2 f. 222v, ignota
- Firenze, Biblioteca Medicea Laurenziana: Plut. 42.23 f. 1r - Plut. 19 dex. 1 f. 7v - Plut. 19 dex. 5 ff. 5v, 6v, 14r, 23v, 55r, 55v, 135v - Plut. 19 dex. 8 f. 5v - Plut. 19 dex. 7 ff. 21v, 89v, 108v - Plut. 30 sin. 3 ff. 42v, 110v, 235v - Conv.Soppr. 321 ff. 145r, 146r, 150v, 151r - Strozzi 146, f. 2r, 12r

– Firenze, Biblioteca Riccardiana: 222 f. 152r - 269 f. 1r - 323 f. 105v - 327 f. 8r - 1422 f. 70r – 829 f. 12r - 1471 f. 43v
– Firenze, Biblioteca Nazionale Centrale: I.III 272-273 f. 32r - C.S D.7.1158 f. 10r - Magl. XII.4, f.17r
– Milano, Biblioteca Ambrosiana: M 76 sup. f. 274
– Pisa, Archivio di Stato: Div. A n. 2 f. 101v
– Siena, Biblioteca Comunale degli Intronati: F.III.3 ff. 2r, 137r.

References

[1] He Sheng and Lambert Schomaker. "Beyond OCR: Multi-faceted understanding of handwritten document characteristics". In: *Pattern Recognition* 63 (Mar. 2017), pp. 321–333. DOI: 10.1016/j.patcog.2016.09.017.

[2] S. He and L. Schomaker. "Co-occurrence Features for Writer Identification". In: *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2016, pp. 78–83. DOI: 10.1109/ICFHR.2016.0027.

[3] Lambert Schomaker, Marco Wiering, and He Sheng. "Junction detection in handwritten documents and its application to writer identification". In: *Pattern Recognition* 48 (June 2015). DOI: 10.1016/j.patcog.2015.05.022.

[4] C. Wick, A. Hartelt, and F. Puppe. "Lyrics Recognition and Syllable Assignment of Medieval Music Manuscripts". In: *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2020, pp. 187–192. DOI: 10.1109/ICFHR2020.2020.00043.

[5] Christoph Wick, Christian Reul, and Frank Puppe. *Calamari - A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition*. 2018. arXiv: 1807.02004 [cs.CV].

[6] Fredrik Wahlberg. "Interpreting the Script: Image Analysis and Machine Learning for Quantitative Studies of Pre-modern Manuscripts". PhD thesis. Acta Universitatis Upsaliensis, 2017.

[7] Francesco Lombardi and Simone Marinai. "Deep Learning for Historical Document Analysis and Recognition - A Survey". In: *J. Imaging* 6.10 (2020), p. 110. URL: https://doi.org/10.3390/jimaging6100110.

[8] K. He et al. "Mask R-CNN". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2980–2988. DOI: 10.1109/ICCV.2017.322.

[9] Zahra Ziran et al. "Text alignment in early printed books combining deep learning and dynamic programming". In: *Pattern Recognit. Lett.* 133 (2020), pp. 109–115.

[10] Samuele Capobianco. "Deep Learning Methods for Document Image Understanding". PhD thesis. University of Florence, 2020.

[11] Kentaro Wada. *labelme: Image Polygonal Annotation with Python*. https://github.com/wkentaro/labelme. 2016.

[12] Yuxin Wu et al. *Detectron2*. https://github.com/facebookresearch/detectron2. 2019.

[13] C. Käding et al. "Fine-Tuning Deep Neural Networks in Continuous Learning Scenarios". In: *ACCV Workshops*. 2016.

[14] Pulkit Agrawal, Ross Girshick, and Jitendra Malik. *Analyzing the Performance of Multilayer Neural Networks for Object Recognition*. 2014. arXiv: 1407.1610 [cs.CV].

[15] R. Girshick et al. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 580–587. DOI: 10.1109/CVPR.2014.81.

[16] M. Oquab et al. "Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1717–1724. DOI: 10.1109/CVPR.2014.222.

[17] Steve Branson et al. *Bird Species Categorization Using Pose Normalized Deep Convolutional Nets*. 2014. arXiv: 1406.2952 [cs.CV].

[18] Artem Babenko et al. *Neural Codes for Image Retrieval*. 2014. arXiv: 1404.1777 [cs.CV].

[19] Yuxin Wu et al. *Detectron2 Model Zoo and Baselines*. 2020. URL: https://github.com/facebookresearch/detectron2/blob/master/MODEL_ZOO.md.

[20] Kaiming He, Ross Girshick, and Piotr Dollár. *Rethinking ImageNet Pre-training*. 2018. arXiv: 1811.08883 [cs.CV].