Detecting and Deciphering Damaged Medieval Armenian Inscriptions Using YOLO and Vision Transformers

Chahan Vidal-Gorène, Aliénor Decours-Perez

Detecting and Deciphering Damaged Medieval Armenian Inscriptions using YOLO and Vision Transformers

Chahan Vidal-Gorène[1,2] [0000−0003−1567−6508] and Aliénor Decours-Perez[2]

[1] École nationale des chartes, Université Paris, Sciences & Lettres, Paris, France
[2] Calfa, Paris, France

Abstract. This paper investigates the development and assessment of a methodology for the automatic detection and interpretation of damaged medieval Armenian inscriptions and graffiti. The research utilizes a newly compiled dataset of 150 images that include a variety of inscriptions, mosaics, and graffiti. These images are sourced from general archaeological site views and vary in quality and type, including drone and archival photos, to replicate real-world database challenges. The results highlight the efficiency of a two-step detection and classification pipeline. The detection phase employs a YOLO v8 model to identify the location and content of inscriptions, achieving an average Precision and Recall of 0.91 and 0.88, respectively. The classification phase uses a Vision Transformer (ViT) to identify similar characters, which outperforms classic CNN-based Siamese networks to handle such a complexity and variation. This approach demonstrates potential for analyzing under-resourced and damaged corpora, thus facilitating the study of deteriorated inscriptions in a variety of contexts.

Keywords: Armenian inscriptions · Digital epigraphy · Computational Paleography · Vision Transformer · Object Detection · Instance Segmentation · Image similarity

1 Introduction

Armenian religious monuments are generally substantially covered with epigraphs and graffiti. They are the writting evidences of the foundation or the renovation of the building, and provide information on the date of construction, the name of the patron, or the purpose of the monument, as well as the passage of the pilgrims. Their deciphering is hindered by the test of time or wilful human damage, which reveals complex even for epigraphists. In some extreme cases, parts or all of the inscriptions are missing and the strong ambiguity of the remaining graphic forms calls for speculation. The most ancient inscriptions that have been dated, prior to the 8th century, though very few, have been the most documented, and can be found in most medieval corpora published since the 19th century, notably in Ališan [2], Yovsēp'ean [24], Kouymjian & Stone [20], Greenwood [8], and Mouraviev [14]. However, the later inscriptions and graffiti, despite partial

collection, identification, and transcription efforts since 1966 [4], are often only briefly documented and deciphered due to their sheer volume. The multiplication of image databases, for example within collaborative projects like Wikimedia-Commons, institutional ones like MonumentWatch, or photogrametric projects carried out by researchers [13] or by private initiatives (Iconem, TUMO), is contributing to the preservation of these testimonies. However, because of their volume, the variety and quality of their formats (written, photographic, 3D), and the damage they have suffered, in the end, very little analysis is done, and the inscriptions remain therefore inaccessible.

This overall inaccessibility is also due to the specific features of the Armenian epigraphic tradition, which encompasses a previous state of the language (or Classical Armenian, even Middle Armenian for some inscriptions), a combination of letters specific to each lapicide – hence, the large variety of monograms –, and the very location of the inscriptions on the monument, which are often high up or poorly oriented (either from the outset or following restoration) and therefore out of reach of the human eye, whether specialist or not (see *infra* Section 3). To this day, no computational approach has yet been explored for Armenian epigraphy, therefore, the aim of this paper is first to explore the feasability of automatic detection of Armenian inscriptions and graffiti in images of varying resolution, and then to explore the classification of the glyphs detected, with a view to proposing an aid for reading these witnesses. This article is an opportunity to build up a first small dataset representative of the Armenian epigraphic production, limited to a capital script from the *erkat'agir* type. This study is set in the context of limited data.

2 Related Works

As a matter of fact, the main difficulty encountered upon computational processing of Armenian sources is the critical lack of data, often incompatible with the training of ML models. If the data creation and retrieval chains for HTR issues regarding medieval handwritten sources are now well established, it is not yet the case for the palaeographical and epigraphical issues, all the more so when sources are damaged. A recent study on Aramaic inscriptions [1] demonstrated that using Generative Adversarial Networks (GANs) to generate 250,000 damaged samples can simulate a representative dataset. This approach achieved over 95% classification accuracy with a common ResNet, given a limited number of different classes (22 Aramaic characters). Although GANs have proven useful, their effectiveness remains primarily within in-domain studies that can tolerate potential overfitting[1, 21]. Generally, these datasets do not exceed a few tens of thousands of samples. In our study, we did not use GANs for data improvement, focusing instead on real-world image variations.

Although direct HTR-type approaches have been tried out, with for exemple the use of Tesseract for the recognition of Tamul inscriptions [7], the state-of-the-art shows today the predominance of a two-step approach: a first step for the character detection and a second for their classification [23]. Predictably, the

detection step is performed as an object detection task by faster-RCNN [18] or YOLO [18, 5], with an Accuracy and a Recall in average above 90% for Ancient and Byzantine Greek. However, both approaches remain unstable, easily producing numerous false positives when met with an advanced state of deterioration, with strong ambiguity of the residual forms [18]. To resolve this ambiguity when the inscription has a high density, the use of a U-net – trained to segment semantically at pixel level – seems to achieve promising results on bones engraved with Chinese inscriptions [6]. The same applies for a CNN-Siamese approach based once again on a faster-RCNN [18]. The assessment remains qualitative for the time being. In turn, the classification stage remains largely prospective: a semi-supervised classification predominates, based on the use of a ResNet [1, 5] or a VGG [22, 10] pre-trained on ImageNet, from which the classification layer is removed in order to cluster the extracted features. This method is not only used to classify characters, but also to date inscriptions (with 85.94% Accuracy for Sinhalese incriptions), to identify Latin [11] and Armenian [22] scripts, or categorise hieroglyphic artifacts [9]. The use of CNN-Siamese networks is also used in Ancient Greek [17] and in Aramaic, that displays a 77% efficient classification, just as a VisionTransformer [15]. These approaches, who are based on similarity, exceed a directly supervised classification [18]. Finally, the detection and partial classification of a set of characters can enable statistical completion of the inscription, which has been used for the first time in Greek with 62% Accuracy and in Latin [3, 12].

3 Notions of Armenian Epigraphy

The Armenian alphabet, dating back from the 5th century, is composed of 36 letters (24 letters from the Greek alphabet among which are interspersed letters to cover sounds specific to Armenian), with the addition of two new letters in the 12th century to cover two new sounds /f/ and /o/. The Armenian epigraphic production has largely remained confined to the *erkat'agir* type, a capital bicameral script generally associated with the uncial [20], that is classicaly used in lapidary inscriptions, ancient graffiti and manuscripts up to the 10-11th centuries – date from which its use in manuscripts declines. The alphabet is also used to write numbers.

The Armenian inscriptions are not very difficult to read: the letters are of substantial size, and if there is no spaces in between words and no apparent logic to the word breaks by default, all letters are evenly spaced out at a width equivalent to an upstroke. Writing lines are sometimes even perceptible. The inscriptions can be V-shaped or flat-bottomed engraved (the latter is more represented in inscriptions located up high on the monument). Some churches are heavily covered of inscriptions, sometimes contiguous with no semantic continuity. The Armenian characters however are composed of a small number of structural features, aside from round letters, they are limited to an upstroke, a curved or round stroke and a connecting or cross stroke. Only the equivalents of the apex and the ending stroke can be used to determine the orientation of the

upstroke or of the round stroke (voir Figure 1). Deterioration of any kind to the inscription on either side of the median line results in extreme ambiguity (e.g. addition of artifacts that can be confused with the round or cross strokes, or even the disappearance of a round or cross stroke critical to decipher the letter), that can only be resolved through the understanding of the context. Depending on the lapicide, the serif of the structural strokes can be enhanced with aesthetic features (e.g. triangular engraving).

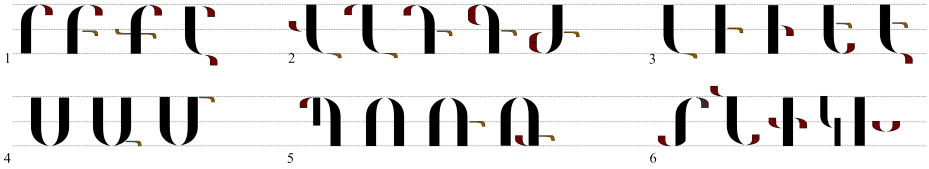

Fig. 1. Schematic representation of structural features of the main Armenian letters, inspired by the model of the inscription of Mastara (7th c.). The first group is composed of one ascender, one right curved stroke and one right cross stroke; the second group is composed of one upstroke, one left curved stroke and one right cross stroke; the third group is composed of one descender, one right curved stroke and one right cross stroke; the fourth and fifth group are composed of two upstrokes (one ascender and one descender), or possibly of one left upstroke and one right cross stroke; and the sixth group is composed of one upstroke and two curved strokes on each side.

Another difficulty lies in the frequent use of ligatures by the lapicide, who is joining together one or more vowels with one or more consonants for space-saving purposes. The defining structural stroke for an Armenian letter being the upstroke, it can indeed be used as a basis for one or several characters.

The most common ligature shape encloses two letters, but there is no structural limitation to the number of possible combinations. Although, for some combinations, the ligature is easy to read for the epigraphist, for other combinations, it is less self evident and it creates an object detection problem in its own right: should a separate distinct class be created for each ligature? The variety and the low representation of each combination do not support this solution. Conversely, should the annotation strictly follow the 36+2 Armenian letters? The overlapping of bounding boxes (bboxes) and the presence of multiple valid interpretations can cause a high false-positive rate. Moreover, the geometric complexity of ligatures and the independent reading order of the signs add to the challenge (see Figure 2).

Figure 2 presents two types of common ligatures: the first one consists in the adjunction of the tops and bottoms of the upstrokes with no impact to the reading direction. The strict annotation per character is possible, but requires the blue bbox to have a shorter height, a format likely to be under represented, for which it is reasonable to assume that the model will tend to detect the whole height of the letter, that will become ambiguous. The second example is more frequent: a single upstroke combining all curved and cross strokes. As a result,

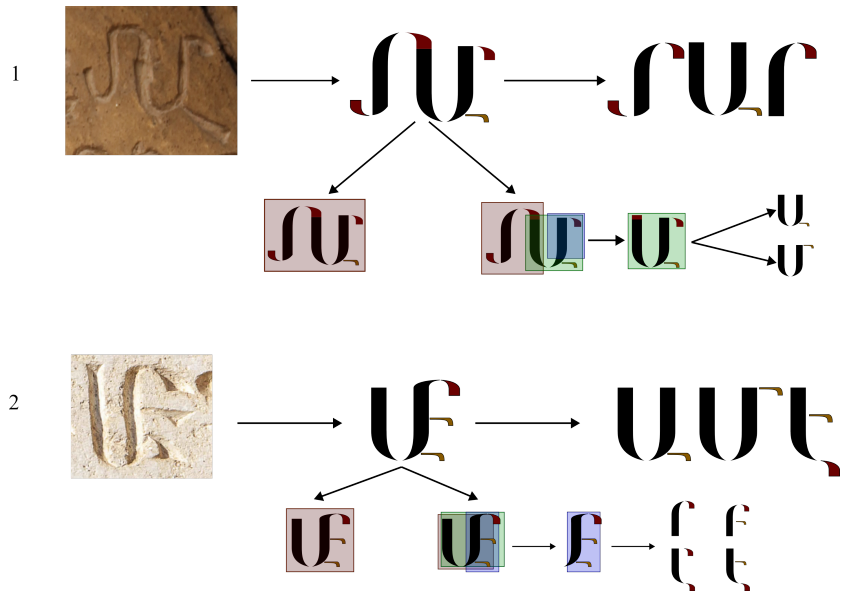

Fig. 2. Two examples of ligatures, their reading, possible annotations and the consequences classification-wise. (1) Inscription of Arudj (8th c.) et (2) inscription of Yeritsmangants (1691).

the green and red bboxes have an intersection close to 1, and the blue bbox may correspond to 4 different shapes, depending on the thickness given by the lapicide to the connecting strokes and serif.

As for the punctuation and the accentuation marks, there is less volume and variety than in Armenian manuscripts. Three types of points are generally found: the median and upper point that indicate a pause, and the two points that marks the ending of the sentence. They come either in diamond or triangular shape, with rectilinear V-shaped engraving. They can be confused with the serif of some characters (orange and red strokes in Figure 1, when the connecting strokes are fine), or with impacts on the stone.

The medieval graffiti display the same attributes as the inscriptions, with a shallower engraving – the stone being usually scratched –, less regularity in the character morphology, and less rectilinear text. They are often limited to an isolated surname, but can also be part of a dedication such as "Remember me [surname]". Medieval graffiti are scarcely found on the walls of medieval churches (also due to the test of time and reconstructions), on the other hand, walls are covered in very recent graffiti of interest not targeted by the present study.

4 Methodology and Dataset

In order to detect and read the damaged inscriptions and graffiti we are setting a two-step pipeline: first, detection of the insription and of the characters; then, a classification of the crops of caractères obtained via similarity calculation.

4.1 Dataset

The dataset is composed of 150 images of inscriptions, mosaics and various graffiti. The stress is put on the inscriptions prior to the 9th century and deteriorated, hard to read, as well as Armenian graffiti from Sinai from the 5-7th centuries [16, 19]. Around a quarter of the inscriptions are dated from after the 9th century, with an upper limit in the 17th century, in order to have more legible images. The images used are not all centered on the inscriptions but are general views of walls or overall views of an archaeological site, and mix recent views with archive photos. The goal of this variety is to simulate a real-life application, with a wide range of shots and qualities (brightness, zoom, sharpness, camera, etc.), representative of existing digital databases (see Figure 3). For example, it is the case of images 3 and 5 in Figure 3, shot in the 1970s, or conversely, image 6 is a 4K view taken by drone to create a photogrammetric model. This nevertheless represents a very significant bias in the ability of the models to converge, especially given the volume considered in the article.

Figure 3 also underlines the difficulty encountered by the dataset with regard to the ambiguity that exists between a deterioration mark and a character (e.g. images 2 and 7), and among graffiti, when mixed with other non-Armenian graffiti (e.g. image 5, where Greek and Latin graffiti are not tagged, like in image 6 if non-Armenian inscription or unrelevant graffiti are present in the overall view).

Table 1 summarizes the distribution of images according to the source type. Mosaics, fairly scarce in the Armenian production, are largely minority within the dataset. There are perhaps as many inscriptions as graffiti, but the count overall is clearly in favor of lapidary inscriptions, the latter generally have more than 3 lines each and more text.

Table 1. Dataset summary

Source	**Type**	**Date**	**Damage**	**Im. Quality**	**Script**	**Insc.**	**Lines**	**Char.**
Inscription	Mixed (drone, camera, book)	7-17th c.	Partial	Mixed	Erkat.	92	303	6,072
Mosaïc	Camera	5-6th c.	No	NTR	Erkat.	9	10	352
Graffiti	Archives	5-6th c.	Yes	Blur, overexposure	Erkat.	74	145	870
TOTAL: 150 images including						175	458	7,294

At the annotation level, we decided to resolve the ambiguity of ligatures mentioned in Section 3 by identifying a type ligature as a separate class (each

Fig. 3. Dataset samples. (1) Door inscription from Aruj monastery (7th c.), (2) Komitas inscription (618 A.D.) from Album of Armenian Paleography, (3 and 5) graffiti of Sinai (5th c.) from The Rock Inscriptions Project, (4) Inscription of Grigoras from Mastara (7th c.), (6) West front of Mastara (7th c.), (7) Yakovb inscription of Ereruyk (6-7th c.) and (8) Uxtatur inscription from Talin (683 A.D.)

different ligature is now the subject of a separate class). Predictably, the dataset is very unbalanced in terms of character classes: the letter *a*, the most frequent in the alphabet, covers 26% of annotation on its own, whereas the ligatures and the characters $\bar{o}$ and *f* have only a single occurrence, and the characters *č*, *ž*, and *š* have fewer than 10 occurrences.

4.2 Step 1: Three-Stage YOLO-Based Inscription Detection

At the detection level, we are evaluating the use of YOLO v8, through the combination of three models:

1. m-Inst$_{\text{insc}}$: detection of inscriptions in an image (instance segmentation, for management of various viewing angles and separation from adjacent inscriptions). The output of this model will correspond to the input of the following models;
2. m-Obj$_{\text{char}}$: detection of glyphs through an object detection approach with a single class (there is not enough representatives per class for direct detection);

3. m-Inst$_{\text{line}}$: detection of a line of text to re-assemble and order glyphs (instance segmentation). Line characters are sorted using the centroid, with simple left-to-right applied to characters, up-and-down applied to lines.

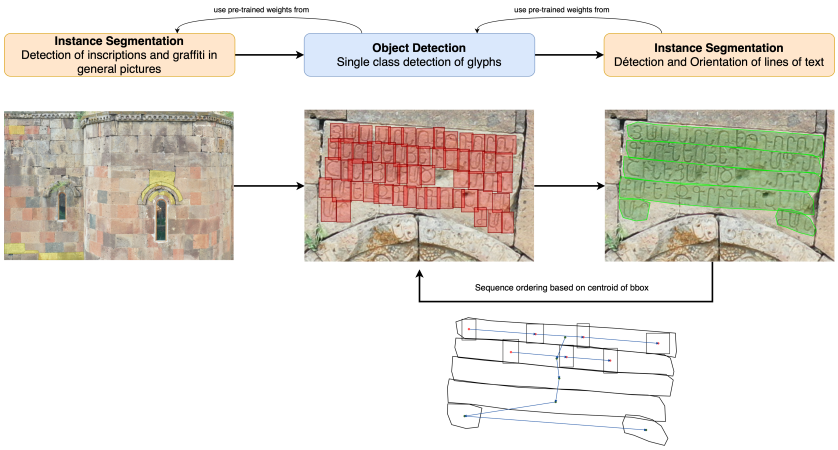

Fig. 4. Detection pipeline involving three YOLO v8 models

All three models perform a single-class classification. For all three, intense dynamic data augmentation is performed during training, similar to that used for papyri detection [18], including image scaling at each iteration and random pixel dropout to simulate further deteroriation. The dropout is set at 20% and the size of image at 1024px. We use the weights from the previous model as the initial weights for the next model, effectively fine-tuning each subsequent model based on the trained weights of its predecessor. This approach aims to incrementally improve performance by leveraging the learning from earlier stages. At each step, 10 cross-validation are performed by redistributing the data into train and val, and then evaluation occurs on the same fixed test set. The scores presented in Table 2 are the average obtained of these 10 experiences.

4.3 Step 2: Classification for Characters Similarity

At the classification level, we are following the approaches used for the Aramaic bowl [15], though changing the task performed and the depth of the models.

The first experiment consists in training a Siamese network to identify pairs of similar images (see Figure 5). The architecture of our Siamese network consists of three main components: a pretrained ResNet50 that will encode an embedding, an euclidian distance for evaluating the similarity, and the contrastive loss function that penalizes the model if the distance between similar inputs exceeds 0.5 or if it falls below this threshold for dissimilar inputs, thus promoting the generation of appropriate embeddings. One of the models of the

Siamese network is trained with similar pairs and the other with dissimilar pairs. The contrastive loss function computes the loss for each pair of embeddings, encouraging similar pairs to have a smaller distance and dissimilar pairs to have a larger distance. The general formula for contrastive loss is as follows: $L = (1-y) \times Dist^2_{Eucl} + y \times max(0, \sigma - Dist_{Eucl})^2$, with $y \in \{0, 1\}$, 0 for similar and 1 for dissimilar, and σ the threshold for dissimilarity.

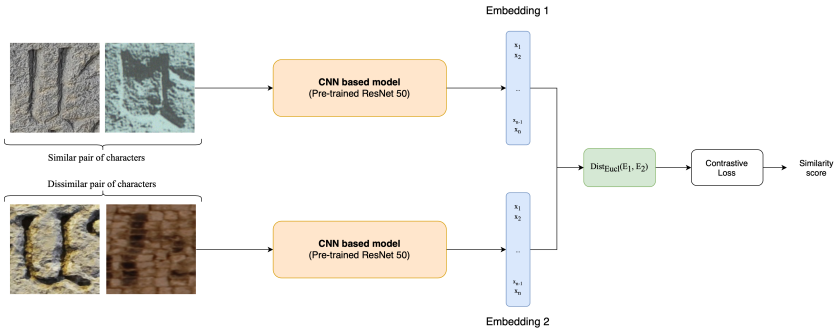

Fig. 5. CNN-based Siamese network using a contrastive loss for similarity classification of characters

The second experiment relies on the use of a distilled Vision Transformer (Data-efficient Image Transformers or DeiT). The approach is no longer based on CNNs but on a full transformer based approach. Images are divided into fixed-size, non-overlapping patches to input into the Transformer encoder. These patches are linearly embedded, and a class token is added as a global image representation for classification. Additionally, absolute position embeddings are incorporated, and the assembled vector sequence is processed by a standard Transformer encoder. Here, the Transformer model's attention mechanism is designed to capture global dependencies and modeling long-range interactions between image patches. The aim is not to train the model with similar and dissimilar pairs anymore, but to create an index from the embeddings of all datasets, and then to perform a similarity search for a given input (see Figure 6).

4.4 Metrics

To measure the effectiveness of the entire pipeline for detection and reading, we define a comprehensive metric that combines the individual evaluation metrics (Precision, Recall, and mean Average Precision, mAP) from each step into a single score. The F1-Score ($F1_i$) at each stage balances Precision and Recall, providing a robust measure that accounts for both true positive detection and the minimization of missed relevant data. The mean Average Precision (mAP_i) evaluates the model's precision at different Intersection over Union (IOU) thresholds, reflecting its consistency and accuracy.

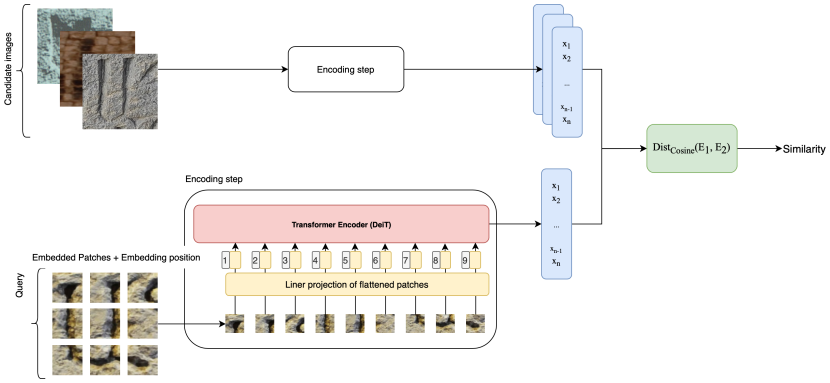

Fig. 6. Vision Transformer using embedded patches and DeiT encoder to perform similarity

Weighting factors (w_i) are assigned to each stage to reflect their relative importance in the pipeline. For instance, we prioritize the detection of inscriptions and characters over lines. We combine these metrics with a weighted sum, balancing the F1-scores and mAPs according to their relevance in the model type. This weight is noted α_i. The final score, Global Score (GS), is the sum of each weighted score (S_i). Thus, for $i \in [1, n]$:

$$GS = \sum_{i=1}^{n} w_i \times S_i = \sum_{i=1}^{n} w_i \times (\alpha_i \times F1_i + (1 - \alpha_i) \times mAP_i) \quad (1)$$

5 Results and Discussion

Table 2 summarizes the outcomes achieved for each detection step. Several configurations have been tested: (i) with or without data augmentation, (ii) single and multi-class for the character detection, and (iii) with or without the use of the output obtained after step 1 (inscription detection).

We observe that data augmentation has a highly variable impact on improving results. In general, it is worth mentioning that it increases the Precision, with a limited effect on the Recall (thus producing more false positives), except for the single-class configuration for character detection (standalone), where data augmentation results in decreased both Precision and Recall (from 0.87 to 0.73 in Precision and from 0.54 to 0.46 in Recall), which could suggest that the augmentation method used may not be optimal or the model overfits without it.

At the character and line detection level, using the output obtained after step 1 (inscription detection) results in a largely better detection, around +30%, even if it is not yet optimal, and the mask produced tends to cut the edges of the inscriptions, thus cropping some of the letters.

Regarding the character detection, we get mixed results in multi-class detection, to be put in perspective with the classification task. Nevertheless, the

Table 2. Inscription detection results using several data augmentation and multi/single-class configuration. Mean Precision, Recall and mAP after 10 random splits

	Augment	P	R	mAP
Inscription detection (mask)				
m-Inst$_{\text{insc}}$ 1	-	0.72	0.87	0.73
m-Inst$_{\text{insc}}$ 2	✓	**0.91**	**0.88**	**0.90**
Char detection (bbox)				
m-Obj$_{\text{char}}$ multi-class 1 (stand alone)	-	0.21	0.18	0.16
m-Obj$_{\text{char}}$ multi-class 2 (stand alone)	✓	0.29	0.17	0.24
m-Obj$_{\text{char}}$ multi-class 3 (using m-Inst$_{\text{insc}}$ output)	-	0.43	0.47	0.28
m-Obj$_{\text{char}}$ multi-class 4 (using m-Inst$_{\text{insc}}$ output)	✓	0.57	0.61	0.54
m-Obj$_{\text{char}}$ single-class 1 (stand alone)	-	0.87	0.54	0.77
m-Obj$_{\text{char}}$ single-class 2 (stand alone)	✓	0.73	0.46	0.69
m-Obj$_{\text{char}}$ single-class 3 (using m-Inst$_{\text{insc}}$ output)	-	0.90	0.82	0.89
m-Obj$_{\text{char}}$ single-class 4 (using m-Inst$_{\text{insc}}$ output)	✓	**0.91**	**0.84**	**0.90**
Line detection (mask)				
m-Inst$_{\text{line}}$ single-class 1 (stand alone)	-	0.67	0.34	0.64
m-Inst$_{\text{line}}$ single-class 2 (stand alone)	✓	0.74	0.31	0.65
m-Inst$_{\text{line}}$ single-class 3 (using m-Inst$_{\text{insc}}$ output)	-	0.94	0.91	0.95
m-Inst$_{\text{line}}$ single-class 4 (using m-Inst$_{\text{insc}}$ output)	✓	**0.94**	**0.93**	**0.96**

model m-Obj$_{\text{char}}$ multi-class 4 (using m-Inst$_{\text{insc}}$ output) achieves 0.43 Precision and 0.47 Recall in detection and direct classification of characters, on the same basis as the results obtained on the papyri with the very same model [18], but with five time less data in training (35,597 characters for the papyri vs 7,294). However, the mAP is low. The scores obtained for YOLO multi-class are the average of all classes and therefore reflect only the few most endowed classes (see *infra* 4.1).

As for the results for similar pair identification, in order to overcome the extreme disproportion between classes, we keep only 50 samples per class. The under-resourced classes are artificially augmented through data augmentation (using blur, rotation, chanel dropout, and pixel dropout), and each experiment is repeated 10 times with random sample selection for each class. Table 3 summarizes the results.

Table 3. Classification results. Accuracy corresponds to the performance of the model in training with manually labeled data, where Precision and Recall correspond to the test using crops of the detection step

Model	P	R	Acc$_{\text{similar}}$	Acc$_{\text{dissimilar}}$
Siamese	0.42	0.42	0.54	0.55
ViT	0.67	0.71	0.78	0.81

The results of the Siamese network in this configuration are much lower than those of the Vision Transformer, and even equivalent to those of YOLO in multi-class detection. The limits of the Siamese network can undoubtedly be explained by the very small size of the dataset due to its complexity and it is appropriate, at this stage, not to completely exclude it from the experiments. The ViT, on the other hand, shows an excellent ability to identify similar pairs in under-resourced contexts, including when the damage or support vary a lot. Table4 gives the global score for each pipeline.

Table 4. Global scores for each configuration of Detection and Classification, with the sum of $w_{detection,i} = 0.6$ and $\alpha = 0.3$ to prioritize mAP in detection tasks

Category	w	α	**P**	**R**	**F1**	**mAP**	**Score**
Inscription detection	0.3	0.3	0.91	0.88	0.89	0.90	0.90
Char. Detection single-class	0.25	0.3	0.91	0.84	0.87	0.90	0.89
Char. Detection multi-class	0.45	0.3	0.43	0.47	0.45	0.28	0.33
Line detection	0.05	0.7	0.94	0.93	0.93	0.96	0.94
Siamese	0.2	1.0	0.42	0.42	0.42	0.00	0.42
ViT	0.2	1.0	0.67	0.71	0.69	0.00	0.69
Combined Results							
Ins+CharSingleClass+Line+Siamese							0.62
Ins+CharSingleClass+Line+ViT							**0.68**
Ins+CharMultiClass+Line							0.65

The final results show a slight advance of the YOLO + VisionTransformer configuration for the detection and identification of characters in damaged Armenian inscriptions and graffiti. Direct detection by YOLO also seems to be possible, but the disproportion of classes between YOLO multi-class and ViT skews the comparison.

6 Conclusion

The article proposes and assesses an end-to-end pipeline for detecting and reading Armenian inscriptions in a very under-resourced and damaged context. The detection scores demonstrate the benefit of a three-step detection of inscriptions, characters and lines, with an average Precision of 0.92 and an average Recall of 0.88. The task of character classification, considered as a task of identification of similar/dissimilar images, due to the high ambiguity of the characters and the small size of the dataset, remains prospective but the use of a Vision Transformer outperforms a classic CNN-Siamese network. The Accuracy obtained on the test set is on average 0.79, with a Precision of 0.67 and a Recall of 0.71 in real conditions. For the future, we plan to significantly strengthen the dataset in order to increase the representativeness of under-resourced classes and to increase the versatility of the model in real images. Incorporating a language model could further improve the results by providing contextual understanding of the

inscriptions. For instance, using a pretrained Armenian language model might help disambiguate characters based on surrounding text. This approach is worth exploring in future work to enhance the accuracy and reliability of our detection and classification pipeline.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Aioanei, A.C., Hunziker-Rodewald, R.R., Klein, K.M., Michels, D.L.: Deep aramaic: Towards a synthetic data paradigm enabling machine learning in epigraphy. Plos one **19**(4), e0299297 (2024)
2. Ališan, L.: Ayrarat. S. Łazar, Venise (1890)
3. Assael, Y., Sommerschield, T., Shillingford, B., Bordbar, M., Pavlopoulos, J., Chatzipanagiotou, M., Androutsopoulos, I., Prag, J., de Freitas, N.: Restoring and attributing ancient texts using deep neural networks. Nature **603**(7900), 280–283 (2022)
4. Collective: Diwan hay vimagrut'iwn (= Corpus of Armenian inscriptions). National Academy of Sciences of Armenia, Erevan (1966–2017)
5. Eyharabide, V., Likforman-Sulem, L., Orlandi, L.M., Binoux, A., Rageau, T., Huang, Q., Fiandrotti, A., Caseau, B., Bloch, I.: Study of historical byzantine seal images: the bhai project for computer-based sigillography. In: Proceedings of the 7th International Workshop on Historical Document Imaging and Processing. p. 49–54. HIP '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3604951.3605523
6. Fu, X., Zhou, R., Yang, X., Li, C.: Detecting oracle bone inscriptions via pseudo-category labels. Heritage Science **12**(1), 107 (2024). https://doi.org/10.1186/s40494-024-01221-5
7. Giridhar, L., Dharani, A., Guruviah, V.: A novel approach to ocr using image recognition based classification for ancient tamil inscriptions in temples. arXiv preprint arXiv:1907.04917 (2019)
8. Greenwood, T.W.: A Corpus of Early Medieval Armenian Inscriptions. Dumbarton Oaks Papers **58**, 27–91 (2004)
9. Hayon, O., Münger, S., Shimshoni, I., Tal, A.: Arcaid: Analysis of archaeological artifacts using drawings. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 7264–7274 (2024)
10. Heenkenda, H., Fernando, T.: Chronological attribution of sinhalese inscriptions using deep learning approaches. Journal of the National Science Foundation of Sri Lanka (2023). https://doi.org/10.4038/jnsfsr.v51i3.11200
11. Kestemont, M., Christlein, V., Stutzmann, D.: Artificial paleography: computational approaches to identifying script types in medieval manuscripts. Speculum **92**(S1), S86–S109 (2017)
12. Locaputo, A., Portelli, B., Colombi, E., Serra, G., et al.: Filling the lacunae in ancient latin inscriptions. In: IRCDL. pp. 68–76 (2023)
13. Magarditchian, A., Vidal-Gorène, C.: L'apport de la photogrammétrie à des prospections archéologiques et paléographiques en Arménie. Études arméniennes contemporaines **14**, 163–183 (2022)

14. Mouraviev, S.: Erkataguir: ou Comment naquit l'alphabet arménien. Academia Verlag, Sankt Augustin (2010)
15. Naamneh, S., Atamni, N., Madi, B., Vasyutinsky Shapira, D., Rabaev, I.R., El-Sana, J., Boardman, S.: Classifying the scripts of aramaic incantation bowls. In: Proceedings of the 7th International Workshop on Historical Document Imaging and Processing. p. 55–60. HIP '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3604951.3605510
16. Negev, A.: The inscriptions of wadi haggag, sinai. Qedem **6**, 1–100 (1977)
17. Sajjad, H., Siddiqi, I., Moetesum, M., Marthot-Santaniello, I.: Learning structural similarities from handwriting on papyri - an application to scribe characterization. In: 2023 International Conference on Frontiers of Information Technology (FIT). pp. 31–36 (2023). https://doi.org/10.1109/FIT60620.2023.00016
18. Seuret, M., Marthot-Santaniello, I., White, S.A., Serbaeva Saraogi, O., Agolli, S., Carrière, G., Rodriguez-Salas, D., Christlein, V.: Icdar 2023 competition on detection and recognition of greek letters on papyri. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) Document Analysis and Recognition - ICDAR 2023. pp. 498–507. Springer Nature Switzerland, Cham (2023)
19. Stone, M.: Armenian Inscriptions from Sinai: Intermediate Report with Notes on Georgian and Nabatean Inscriptions. Harvard Armenian texts and studies, Maitland Publications (1979)
20. Stone, M.E., Kouymjian, D., Lehmann, H.J.: Album of Armenian paleography. Aarhus University Press, Aarhus (2002)
21. Vidal-Gorène, C., Camps, J.B., Clérice, T.: Synthetic lines from historical manuscripts: an experiment using gan and style transfer. In: International Conference on Image Analysis and Processing. pp. 477–488. Springer (2023)
22. Vidal-Gorène, C., Decours-Perez, A.: A Computational Approach of Armenian Paleography. In: Barney Smith, E.H., Pal, U. (eds.) Document Analysis and Recognition – ICDAR 2021 Workshops. pp. 295–305. Lecture Notes in Computer Science, Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-86159-9_20
23. Vijayalakshmi, R., Gnanasekar, J.: A review on character recognition and information retrieval from ancient inscriptions. In: 2022 8th International Conference on Smart Structures and Systems (ICSSS). pp. 1–7 (2022). https://doi.org/10.1109/ICSSS54381.2022.9782241
24. Yovsēp'ean, G.: K'artēz hay hnagrut'ean (= Armenian Paleography Atlas). Šołakat' **1**, 170–214 (1913)

7 Appendix

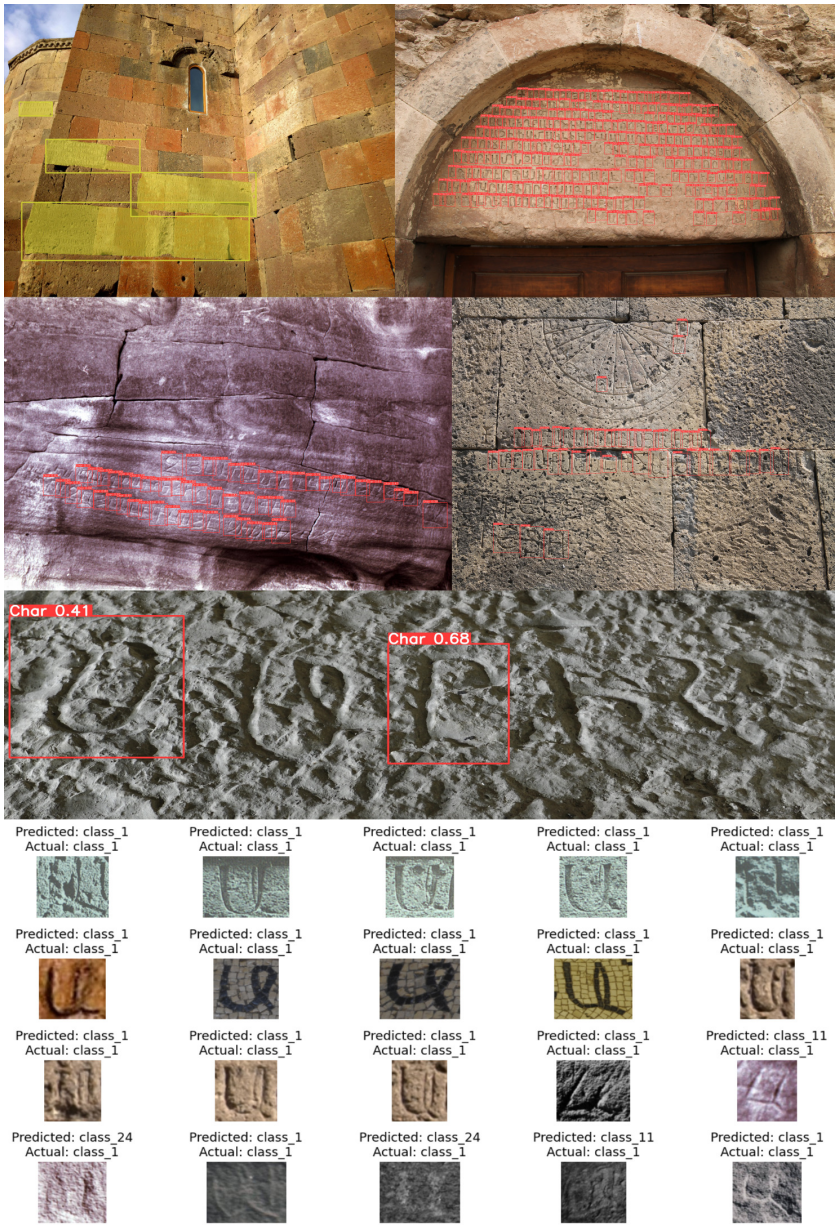

Fig. 7. Qualitative results of inscription detection, single-class character detection and character similrity using ViT