International Journal of Advanced Technology and Engineering Exploration, Vol 10(109)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2023.10102331

Research Article

Deciphering the past: enhancing Assyrian Cuneiform recognition with YOLOv8 object detection

Elaf A. Saeed[1*], Ammar D. Jasim[1] and Munther A. Abdul Malik[2]
Department of System Engineering, College of Information Engineering, AL-Nahrain University, Baghdad, Iraq[1]
Department of History, College of Literature, Baghdad University, Baghdad, Iraq[2]

Received: 28-September-2023; Revised: 20-December-2023; Accepted: 22-December-2023

Abstract
Cuneiform writing offers insight into our distant past. Emerging in the latter part of the 4th millennium BCE, cuneiform script is among the earliest known writing systems, alongside Egyptian hieroglyphs. It is believed to have originated with the Sumerians in southern Mesopotamia. Used for nearly three thousand years, it was eventually replaced by more accessible alphabet-based systems. Cuneiform texts were inscribed on various materials, but clay tablets were preferred due to their availability. Over 500,000 cuneiform documents have been found, with many yet to be analyzed by philologists. This highlights the need for effective methods to study the extensive cuneiform writings, traditionally examined manually. Deciphering ancient tablets is time-consuming, requiring extensive expertise. Signs on Assyrian cuneiform tablets were aimed to be detected in this study using the YOLOv8 object detection pretraining model. About 900 images of Assyrian tablets from the Iraq Museum were compiled and expanded to over 2000 through preprocessing and augmentation. This led to the identification of 11 new Assyrian references, with a mean average precision (mAP) at 50% of 82.7%, a precision of 71.3%, and a recall of 85.6% being achieved. The detection of cuneiform signs, as well as the selection and pronunciation of the modern Assyrian dialect, was facilitated by this research, aiding researchers in reading with a pre-trained model.

Keywords
Cuneiform writing, YOLOv8, Assyrian tablets, Philological analysis, Ancient script decipherment.

1.Introduction
Writing systems were developed in Mesopotamia in approximately 3200 BCE [1]. The writing system was initially based on symbols that express things, which are found in the Sumerian language. The writing used in the Assyrian and Babylonian languages appeared after symbolic writing went through many stages of development [2]. Compared to the hieroglyphic visual language, the cuneiform system is more verbal and expressive and uses distinct terminology to express specific meanings. Many cuneiform tablets have been discovered, amounting to more than 10,000 tablets located in the International Museum and the Iraqi Museum, which contains approximately 2,000 tablets [3, 4]. Cuneiform in Mesopotamia developed into the Assyrian cuneiform language. Writing became read from right to left, and symbols were engraved on stone or clay tablets.

Around 600 letters in the cuneiform alphabet, each comprising one or more symbols. These wedge-shaped symbols are arranged either horizontally, vertically, or diagonally. As seen in *Figure 1*, the letters and the symbols that go with them differ from one character to the next in terms of their number, placement, and orientation taken from the Museum of Iraq [5, 6].

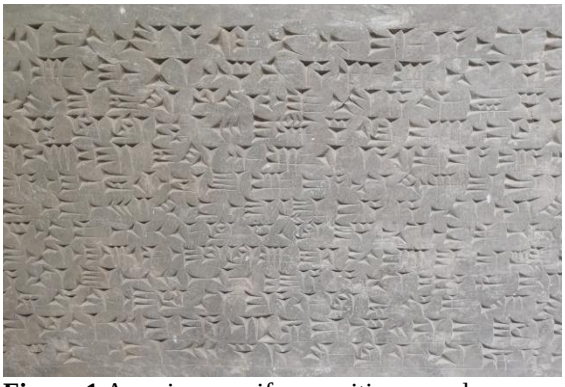


Figure 1 Assyrian cuneiform writing samples

*Author for correspondence

There are very few translations of the cuneiform language, so this problem must be solved using modern technological techniques. However, deciphering ancient clay tablets takes time, calling for years of expertise. Therefore, many researchers have worked in this field using different techniques and strategies. Computer vision is a commonly used and quickly developing technique for locating objects in images. The effective utilization of machine learning techniques for word spotting on cuneiform datasets is typically constrained by the need for more annotated training samples. Due to the substantial volume of data needed, manually generating it is a laborious operation for both two-dimensional (2D) and three-dimensional (3D) datasets [7, 8]. Identifying individual characters is not problematic in most writing systems; however, recognizing cuneiform signs poses challenges for many reasons. Whitespace between characters in many scripts greatly facilitates detection by enabling the separation of localization and classification into two sequential phases. Nevertheless, cuneiform signs are frequently engraved in close proximity to each other, resulting in the need for their placement and classification to be closely linked [9].

To assist Assyriologists in their study, our objective is to streamline the process of deciphering cuneiform scripts. Specifically, our objective is to develop a cuneiform sign detector that provides the location of a sign (included in a bounding box) and identifies its sign pronunciation. Our objective is not to supplant Assyriologists through the automated generation of transliterations or translations but rather to assist them by providing sign identification as an essential component of deciphering cuneiform script.

Within the framework of cuneiform writing, methods that rely on lines encounter difficulties caused by the inclusion of an extra step for line recognition. This is because lines are frequently impaired and challenging to trace accurately due to gaps and misalignments. A word-level method is not suitable since cuneiform signs have several meanings and can be used in various unclear ways to construct a word. in this paper, our methodology employs a character-based strategy, which yields a sign detector that directly produces bounding boxes for each particular cuneiform sign and adding a label represents the pronunciation of the cuneiform sign by using pretraining model. Sign-level bounding boxes are essential for Assyriologists as they aid in understanding the detector's decision-making process [10].

The contribution of this paper is the creation of a dataset, which was achieved by collecting more than 300 images of Neo-Assyrian tablets from the Iraqi Museum. The model was trained to perform detection using you only look once version 8 (YOLOv8). Satisfactory results were achieved, and the Neo-Assyrian signs were well recognized.

In this article, the literature on language detection and the study of written languages using the cuneiform writing system were reviewed in section 2. All methods used for preparation and training were discussed in section 3. Prediction results were presented in section 4. The results were discussed in section 5. The conclusion and future work were outlined in section 6.

2.Literature review

A mechanism has been devised to facilitate the daily responsibilities of Assyrian scholars. To achieve a transformation invariant function, one can calculate the volumes of many concentric spheres that intersect the volume below the surface of the 3D model at each location [12]. The feature vector represents the function due to its multiscale approach. The system, known as GigaMesh, can automatically extract characters by classifying these feature vectors using auto-correlation. It only requires one parameter, which is the approximated line width (wedge) measured in millimeters. The utilization of cutting-edge 3D technology in conjunction with GigaMesh yields a resilient and expeditious workflow for a diverse array of cuneiform panels, particularly those that pose challenges for human processing.

An innovative approach was presented to translating ancient writing by employing artificial neural network (ANN) technology [13]. The multi-layer perceptron (MLP) neural network has effectively converted images of Sumerian cuneiform signs into their corresponding English characters. Nevertheless, there is a limitation on the quantity of data that may be used for photographs. This proposed methodology has achieved an exceptional level of accuracy, reaching a benchmark of 100%. This method enables expedient and precise comprehension of Sumerian cuneiform symbols for explorers and researchers. A minimum error value of 0.0009999 was achieved.

The fundamental constituents were identified, specifically the strokes, of cuneiform symbols depicted in photos of ancient cuneiform tablets [14]. The purpose is to facilitate efficient optical character recognition (OCR) using contemporary computer

vision methods. One notable distinction between our technique and previous methodologies is our utilization of 2D photographs rather than 3D models. This choice is motivated by the abundance of publicly accessible web archives containing a far more significant number of 2D images. Rusakov et al. [15] the software was developed with the purpose of enabling the highlighting of stroke characters through the use of convolutional image filtering techniques. These edge filters are frequently employed to reduce the background of desirable objects and emphasize their edges. The properties of Hough transformation [8], specifically lines and their orientation, were utilized. The neural network architectures YOLOv5 and Detecto were created to accurately identify and locate horizontal strokes in cuneiform tablet pictures that are partitioned into squares measuring 416×416 pixels. The classifier utilizing the Detecto algorithm achieves a commendable accuracy of 90.5% but with a 25% incidence of false positive predictions. Conversely, the classifier employing the YOLOv5 algorithm exhibits a lesser level of accuracy when applied to the cuneiform data.

This study introduces an automated method for creating training samples through the utilization of generative adversarial networks for domain adaptation [15]. This method facilitates the transition between the visual representations of hand-drawn cuneiform autographs and 2D projections of 3D-scanned cuneiform tablets without relying on any classification data. By following this approach, highly favourable outcomes are attained. The findings indicate that the utilization of picture generation enhances the average performance, increasing the mean average precision (mAP) from 85.7% to 89.4%. An increase of 5.9% (from 78.9% to 86.8%) might be attained in the scenario of spotting all tablets. This method is restricted to the transmission of images from one to another.

A deep-learning-based sign detector was presented to accurately identify and categorize cuneiform signs in photographs of clay tablets [16]. A weakly supervised strategy was suggested, which involved aligning tablet images with their associated transliterations in order to identify the location of transliterated signs inside the tablet image. These localized signs were then used instead of annotations to retrain the sign detector. Deep learning necessitates substantial quantities of training data comprising bounding boxes encompassing cuneiform signs, which are not readily accessible and incur significant expenses when procuring the cuneiform script. In order to address this issue, we employ pre-existing transliterations, which provide a detailed depiction of the tablet's content using the Latin script on a sign-by-sign basis. While the method demonstrates some effectiveness in weakly guided scenarios, the performance of the cuneiform sign detector can be significantly improved with a limited amount of annotations. We analyze this detector on a vast collection of clay tablets dating back to the Neo-Assyrian period.

However, in this study, an attribute representation is established using the Gottstein-System [17]. The objective is to break down signs based on wedge typology and facilitate a logical representation for classes of signs while also allowing these representations to be shared among physically comparable cuneiform signs. The purpose of adapting it was to describe wedge expressions and establish the query-by-expression (QbX) retrieval scenario. Our technique can represent questions for an open-set retrieval scenario, which is superior to searches based on sign identifications (IDs). In addition, the Gottstein-System utilizes phrases to indicate wedge crossings or position relations visually. By following this approach, favourable outcomes are attained.

This paper presents novel algorithms for achieving consistent results in the recognition of cuneiform symbols [18]. The algorithms focus on addressing issues such as problem spots and writing lines by utilizing statistical skewness measures, image morphology, and distance transform concepts. Additionally, the algorithms involve selecting an appropriate binarization method and removing writing lines and spots. This study discusses the limitations of iterative approaches when the colour histogram is in the dark interval. It also explores the selection of the thresholding value between the Niblack and Sauvola methods based on the Skewness measure. The issue associated with this proposed algorithm pertains to the chosen threshold value. The less mean square error (MSE) that was reached is 12.62 for 110.89 cuneiform symbols' length.

Various methodologies have been documented for the identification and detection of cuneiform symbols on both 2D and 3D depictions. In 2012, Mara [19] presented a methodology for extracting 2D vector illustrations from 3D cuneiform images. Building upon Howe's part-structured inkball models [20], Bogacz et al. [21] analyzed these spline graphs from [19] in order to align them with cuneiform signs.

They used a similarity metric that relied on graph representations. Additionally, in their study [22], the authors showcased the utilization of spline graphs as structural attributes for pattern matching. Massa et al. [23] pursued a comparable objective and elucidated a technique for extracting graph representations from 2D cuneiform paintings. The authors introduced a method in [22] that utilizes part-structured models to identify cuneiform signs by segmentation. In addition to the graph-based feature representations, Rothacker et al. [24] introduced a technique that utilizes a bag-of-features (BoF) approach relying on SIFT-Descriptors. In this case, the segmentation-free identification of cuneiform signs is achieved by incorporating hidden markov models (HMMs) into a patch-based (sliding window) method.

Rath and Manmatha [25] introduced a popular historical document word detection approach. They used a word segmentation of document pictures. Encoded word pictures included projection profiles and ink backdrop transitions. Dynamic time warping calculated feature representation similarity. Recent breakthroughs in computer vision influenced word-spotting algorithms. These methods usually adapt automatically to the issue domain.

Rusiñol et al. [26] employed BoF representations for segmentation-free word detection. BoF describes local document image areas as orderless feature collections. This method has two key benefits. Unsupervised BoF estimation from sample data. Thus, designing features requires no manual work. This is useful since historical document collections vary in script appearance. Since no script spatial placement assumptions are needed, BoF representations can be used segmentation-free. This is very useful for historical document analysis. Dense writing, inhomogeneous spacings, and document degenerations make word-level heuristic segmentation difficult.

The polygon approximation method was demonstrated as an effective technique for extracting features [27]. It has been compared to the elliptic Fourier descriptor method in recognition tasks and has shown high accuracy results. This was achieved by using a support vector machine classifier with multiple classes and relying on its discriminative functions. The application of this study involves the utilisation of two Datasets. The initial Dataset comprises 320 photographs of cuneiform sign patterns, which are used to assess the most effective method for feature extraction. The second dataset comprises 240 images depicting cuneiform letters, which are used to assess the performance of the recognition system. The training dataset for the agents consists of four 2D triangular patterns.

This research examines the process of automatically transcribing phonological information from transliterated corpora [28]. The phonological transcription of Akkadian offers a linguistically attractive means of representing the language. This transcription is standardized based on the grammatical characteristics of a certain dialect and clearly displays the Akkadian equivalents for Sumerian logograms. Due to the absence of inflection markers for logograms in cuneiform language, the inflected form must be deduced from the context of the phrase. This is the initially recorded endeavour to transcribe Akkadian automatically. By employing a context-aware neural network model, it is possible to achieve near-human performance in automatically transcribing syllabic tokens with a recall rate of 96% at 3. However, transcribing logograms proves to be more complex, with a recall rate of 82% at 3.

Allusion was made to a cohort of studies that employed an identical methodology. Additionally, an alternative iteration of YOLO, along with various other techniques like image processing and computer vision, were employed to successfully extract cuneiform signs, identify them, translate them into different languages, and transcribe their pronunciation. The ideas that have been developed aim to streamline the process of reading and translating cuneiform symbols, with the goal of assisting both scholars and enthusiasts. However, the current level of research still needs to be substantial enough to realize our study objectives fully.

3.Methods

3.1Methodology of Yolov8

Object detection is currently regarded as one of the most challenging and popular areas in computer vision [29, 30]. There are two types of object detectors: two-stage and single-stage [31, 32]. Single-stage detectors aim to identify objects in one step by targeting all spatial region proposal changes with a more accessible architecture. In contrast, two-stage detectors focus on a selected region proposals approach through a more advanced design. Additionally, the introduction of YOLO has significantly increased detection accuracy, often surpassing that of two-stage detectors [33, 34].

In its modified form, the most recent YOLO model, known as YOLOv8, can perform tasks like object detection, image categorization, and instance segmentation. Ultralytics, the creator of the significant and industry-defining YOLOv5 model, also developed YOLOv8. YOLOv8 features many architectural modifications and enhancements over YOLOv5, improving the developer experience. The performance of the object detector is evaluated using inference time and detection accuracy. This study aims to examine various metrics to assess the effectiveness of different YOLO versions and model sizes on the common objects in context (COCO) dataset. YOLOv8 has various sub-versions that differ in speed and the number of parameters. The YOLOv8n (nano) version has fewer parameters and is slower than other versions, such as small (s), medium (m), large (l), and x-large (x).

After replacing the Cross-Stage Partial layer (CSPLayer) with the C2f module, YOLOv8 now has a backbone comparable to YOLOv5. The C2f module, standing for cross-stage partial bottleneck with two convolutions, enhances identification accuracy by merging contextual and high-level features. YOLOv8 introduces an anchor-free model with a decoupled head that processes objectness, classification, and regression tasks independently, allowing each branch to focus on its specific task and thereby improving model accuracy. The output layer of YOLOv8 activates the objectness score using the sigmoid function, which indicates the likelihood of the bounding box containing an object. It also employs the softmax function to represent object probabilities across different classes. For calculating loss, YOLOv8 utilizes binary cross-entropy for classification and combines complete Intersection over Union (CIoU) and Distribution Focal Loss (DFL) for bounding box loss, significantly enhancing object detection, particularly for small objects.

Additionally, the semantic segmentation model YOLOv8-Seg is available. This model, unlike the traditional YOLO architecture, employs a CSPDarknet53 feature extractor and the C2f module as its backbone, followed by two segmentation heads that predict semantic segmentation masks for the input images. Like YOLOv8, it includes five detection modules and a prediction layer, and despite being fast and efficient, YOLOv8-Seg has achieved top scores in object detection and semantic segmentation benchmarks.

YOLOv8 can be executed via the command line interface (CLI) or installed as a Python package manager (PIP) package. It also offers numerous integrations for labeling, training, and deploying, making it a versatile tool for various object detection and image processing tasks.

When tested on the Microsoft common objects in context (MS COCO) dataset test-dev 2017, YOLOv8x achieved an average precision (AP) of 53.9% using a picture size of 640 pixels. This represents an improvement over YOLOv5, which achieved an AP of 50.7% with the same input size. Additionally, YOLOv8x demonstrated a speed of 280 frames per second (FPS) on an NVIDIA A100 with TensorRT.

3.2The architecture of the proposed model

Figure 2 shows the intelligent proposed model using the YOLOv8x pretraining object detection model. The user provides data in labeled images, and pre-processing is done to resize the images to 640×640 px, Auto-orient, Auto-adjust contrast, and Tile. Also, provide augmentation to images (Rotation, Brightness, Mosaic, and Bounding Box Rotation). Data is delivered to the Yolov8x object detection algorithm for training and validation after pre-processing.

The accuracy, precision, and recall of the output from the yolov8 of different versions of split, pre-processing, and augmented algorithms are compared. The two primary results from Yolo are the bounding box and class prediction. A third output has been added: to pronounce the name of the signs (category name) discovered and identified in the image. Every image's predicted class for the objects found will be a string, such as "NU." The Google text-to-speech (gTTS) [44] package can then be used to submit the written description to the Google written-to-speech application programming interface (API). *Figure 3* shows the flowchart for the entire proposed model architecture.

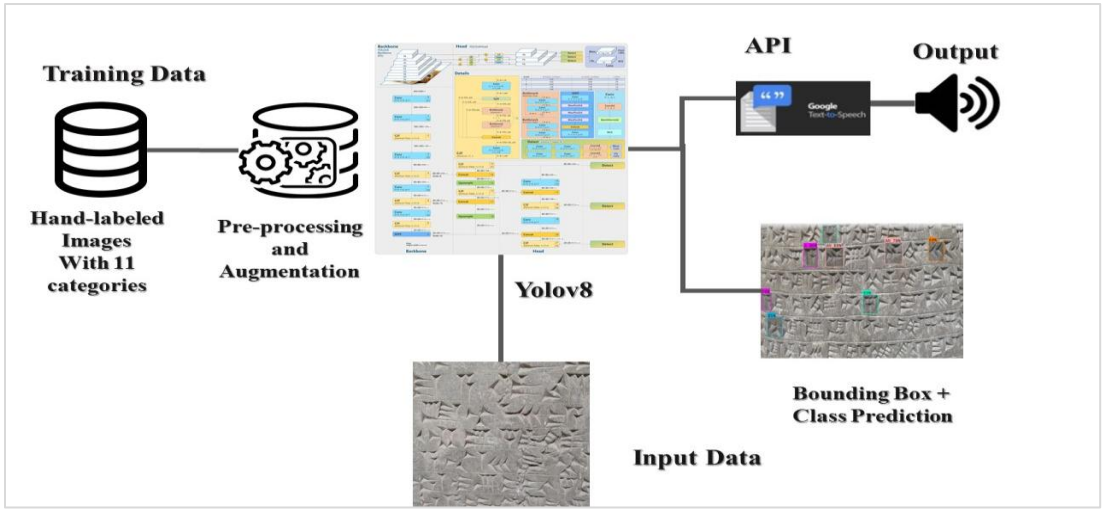


Figure 2 The proposed model block diagram

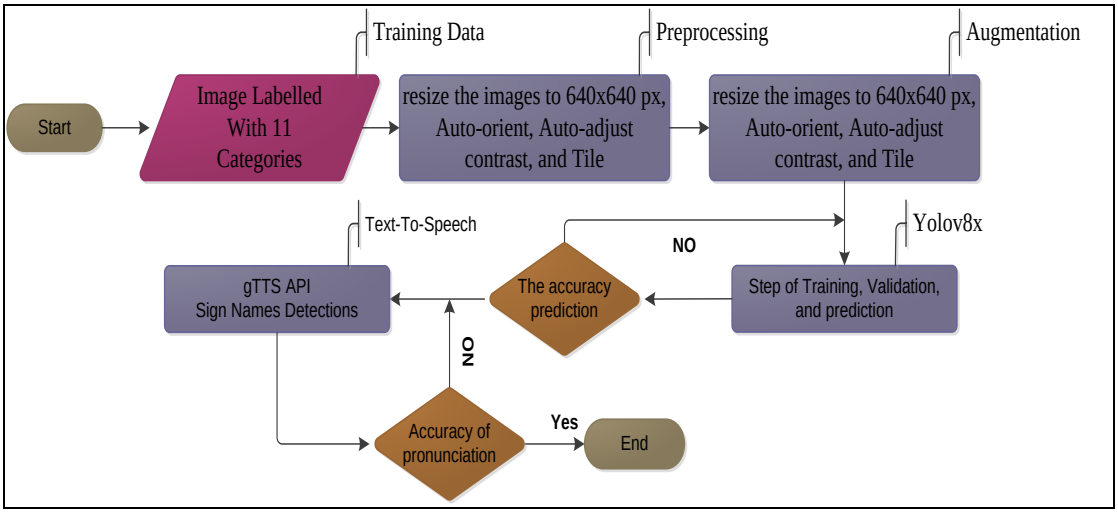


Figure 3 The proposed model flowchart

3.3Challenges of cuneiform sign detection

When working with Yolo, one of the biggest challenges faced is collecting data, which is not available and ready online.

- White space between characters in scripts greatly facilitates detection by enabling the separation of localization and classification into two sequential processes. Nevertheless, cuneiform symbols are frequently engraved without gaps between adjacent symbols, resulting in their localization and classification being interconnected [45].
- The challenge is heightened by the fact that there are over 900 distinct sign code classes with similar characteristics. This is because most cuneiform signs are made up of a small number of wedge shapes (lying, standing, and diagonal) that are arranged in different ways. As a result, many sign code classes have a high degree of similarity to each other. Conversely, cuneiform signs belonging to the same sign code class can exhibit significant differences in their visual representation when written by various scribes (considerable intra-class variance)[28].
- Throughout thousands of years, numerous tablets have been destroyed, which further complicates the discovery of signs. The damage manifests in diverse forms, including surface damage, cracks, holes, and missing components. Frequently, tablets necessitate reassembly from fragmented components[46]
- Cuneiform script is composed of 3D wedges inscribed onto the surface of 3D clay tablets. Due to our reliance on 2D images of clay tablets, the

recognition of individual signals is affected by lighting and visual distortions[47].

- Writing on soft clay tablets results in the distortion of previously written signs due to the influence of nearby signs. The phenomenon of plastic deformation is exclusive to the cuneiform script and adds to the challenge of differentiating between signs, as wedges may become indiscernible during the writing process[45].
- Conventional methods in computer vision and machine learning for object detection necessitate substantial quantities of supervised data. Regarding the identification of signs, this entails the annotation of ten thousand bounding boxes around cuneiform signs. Due to the implicit nature of sign identification performed by Assyriologists, there is a lack of available bounding box annotations[13].

3.4Dataset creation

3.4.1Dataset collection

Many manually annotated input photos were required to train high-quality models. For accurate detection of each class, approximately 1500 images were recommended [48]. A collection of pictures of Assyrian tablets located in the Iraqi Museum was compiled. Pictures were taken of 14 Assyrian tablets, inscribed on stone. The challenges faced during data collection included: firstly, the need for clearer cuneiform signs due to the age and damage of the tablets. Secondly, the small number of tablets found, prompting a search for additional pictures in the Iraqi Museum to improve the dataset. Thirdly, the presence of very small and overlapping cuneiform tablets made labeling difficult. To obtain a larger amount of labeled data, the dataset size was either expanded or data augmentation techniques were employed to enhance the dataset's volume, thereby increasing the model's capacity for accurate identification. *Figure 4* displayed some pictures of the Assyrian tablets.

Figure 4 Some pictures of Neo-Assyrian tablets

3.4.2Dataset labelling

After reviewing the gathered images, the most prevalent indications in the sentences were selected. The form and pronunciation of these 11 Neo-Assyrian signs, which were found to be more common than others, are displayed in *Table 1*. When describing signs, the sign name is written as a label. To facilitate reading, the signs are represented in well-known English letters or other languages. For instance, the sign ( ) is pronounced similarly to A, and the remaining signs ( ) are pronounced similarly to MAN.

Table 1 The Neo-assyrian signs with signs name

S. No.	Neo-assyrian sign	Sign name
1		AN
2		ŠA
3		MAN
4		A
5		Di

6		E
7		EN
8		NA
9		NU
10		BI
11		RI

3.4.3Dataset pre-processing and augmentation

The YOLOv8 complicated solution, built by Ultralytics, can be accessed on the GitHub platform. [49]. The system comprises a proficiently trained neural network with accompanying scripts for training and detecting. The requirement for square input photographs accompanied by labels in .txt format is essential. The dataset was annotated, pre-processed, and augmented by using the Roboflow platform. The dataset consists of 882 labeled images with 5114 annotations. We used the resize image (640×640 pixel), auto-orient, auto-adjust constraint, and tile (2×2) preprocessing. Then, rotation (-15° - +15°), brightness (-25% - +25 %), Mosaic, and bounding box rotation (-15° - +15°) augmentation.

For preprocessing, the auto-orient feature adjusts the pixel direction in the image, leading to altered orientations of objects. This is beneficial for detecting targets in rotated images. Implementing automatic constraints enhances the neural network's ability to understand object nature. Edge detection is a crucial principle in computer vision, applicable to classification, object identification, or segmentation, as it enhances the clarity of edges by amplifying contrasts between neighboring pixels. The Tile(2×2) function splits images into smaller sections, using a 2 rows and 2 columns split to improve accuracy on small objects. Resizing images to 640×640 pixels adapt them to the accepted input size for YOLOv8.

Data augmentation enhances the semantic variety of a dataset. It often involves applying random rotations, altering the object's position within the frame, and is crucial for accurately updating bounding boxes in object detection. Adjusting brightness introduces variations in image luminosity, aiding model adaptability to different lighting conditions and camera setups. Mosaics, created by merging four source photos, help preserve object scale, combine classes for comprehensive training, and vary the number of objects in the images, enhancing the model's performance in complex scenarios.

Bounding box augmentation modifies the content within source image bounding boxes, creating new training data more aligned with the problem's specific conditions. *Figure 5* displays images with their corresponding labels post-training and the results post-validation, illustrating the appearance after preprocessing and augmentation.

3.4.4Dataset splitting

The dataset was split into different ratios (three versions (V) of splitting) of training, validation, and testing, as shown in *Table 2*.

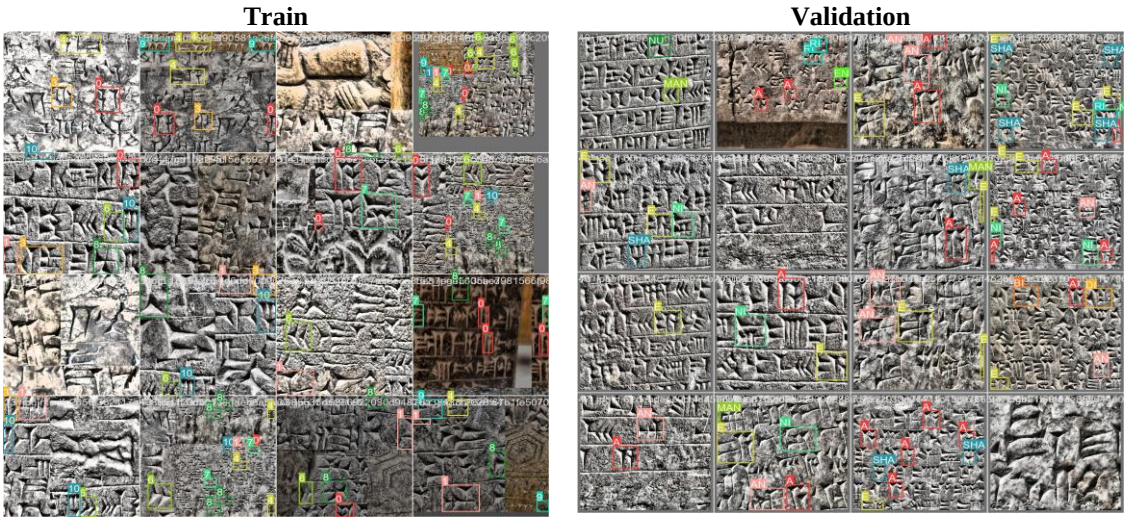


Figure 5 The pictures with the labels that are trained on and validation results

Different ratios were chosen for training and testing. Firstly, this is because the data set is unbalanced. The number of labels for each class is not equal, and there are differences in large numbers. This is only to

improve training and add more images to the few classes. We have a small number of images in relation to the number of classes because in order for the training to be highly accurate, each class must have at least 1500, and this is not available. The number of annotations for each class, A=1,289, SHA=843, NI=575, E=502, AN=496, MAN 438, NU=285, RI=267, EN=224, BI=118, and DI=77.

Table 2 Dataset split

Version No.	Train –val-test	# of all images	Pre-processing	Augmentation
1	2388 - 224 – 116	2728	1- Auto-orient. 2- Resize (640×640 pixel) 3- Auto-adjust contrast.	1- Rotation. 2- Brightness. 3- Mosaic. 4- Bounding Box Rotation.
2	2247 - 87 - 46	2380	1- Tile. 2- Auto-orient. 3- Auto-adjust contrast.	
3	1854 – 176 - 88	2118		

4.Results

4.1Training and evaluation

4.1.1Dataset training

YOLOv8 exclusively employs training scripts designed to shuffle the dataset. To export predicted photographs and labels as files, display them, and use them outside of prediction notebooks, modifying the script is necessary. The YOLOv8 architecture was trained on Google Collaboratory using the Compute Unified Device Architecture (CUDA) programming language for 100 and 50 epochs, respectively. Training occurred on a Tesla T4 Graphical Processing Unit (GPU) with 40 multiprocessors and a total RAM capacity of 15109 MB. The total time taken for training over 100 epochs ranged from 2 to 3.5 hours, while for 50 epochs, the training duration was approximately 1.5 hours.

4.1.2Evaluation

The best outcome for version (V) two was achieved with 50 epochs, yielding a mAP at 50% (mAP50) of 82.7%, a mAP from 50% to 90% (mAP50-90) of 61.3%, a precision of 71.319%, and a recall of 85.647%. The most favorable evaluation results after training are presented in *Table 3*.

Table 3 The best evaluation results after training

V	Epochs	mAP50	mAP50-90	Precision	Recall
1	100	63.6%	45.115%	75.62%	58.66%
2	50	82.7%	61.2%	71.319%	85.647%
3	100	76.5%	53.7%	71.74%	79.412%

The confusion matrix is used to assess the results of a test. It is arranged with predicted classes on the y-axis and actual classes on the x-axis. This matrix determines the model's accuracy by comparing the results obtained after training and testing [50]. For this purpose, platforms such as Google Collaboratory or Kaggle's open Python environments were utilized. The performance of each architectural design was evaluated using standardized metrics such as precision P (Equation 1), recall R (Equation 2), and F-measure F (Equation 3).

$$P = \frac{TP}{TP + FP} \quad (1)$$

$$R = \frac{TP}{TP + FN} \quad (2)$$

$$F1 - score = \frac{2.R.P}{R + P} \quad (3)$$

Where:

True positives (TP) refer to instances where strokes were correctly recognized.

FP = false positives (predicted strokes that don't actually occur).

FN = false negatives (accurate strokes weren't discovered).

The AP metric is built on precision-recall metrics, handles several object categories, and uses intersection over union (IoU) to define a positive prediction.

Several measures are not just crucial for YOLOv8 but also have wide applicability across various object identification models.

- *IoU*: is a metric that calculates the degree of overlap between a predicted bounding box and a ground truth bounding box. It has a crucial function in assessing the precision of object localization.
- *The AP*: The metric calculates the integral of the precision-recall curve, yielding a singular number that summarises the precision and recall performance of the model.
- *mAP*: is a metric that expands on the concept of AP by computing the average AP values across various object classes. This is advantageous in situations when multiple classes of objects need to be detected, as it allows for a thorough assessment of the model's effectiveness.
- *Precision and Recall*: Precision measures the ratio of true positives to all positive predictions, evaluating the model's ability to minimize false

positives. Crucial when reducing the number of false detections is an objective. Conversely, Recall quantifies the ratio of correctly identified positive cases out of all the actual positive instances, assessing the model's capability to identify all occurrences of a specific class.

- *The F1-score* is calculated as the harmonic mean of precision and recall, offering a well-balanced evaluation of a model's performance by considering both false positives and false negatives.

mAP50 refers to the mAP, which is computed by evaluating the IoU at a threshold of 0.50. It quantifies the accuracy of the model by solely evaluating the detections that are deemed "easy." mAP50-95 refers to the average value of the mAP, which is computed by considering several IoU thresholds ranging from 0.50 to 0.95. It provides a thorough perspective on the model's performance across various levels of detection difficulty.

These measurements provide information about precision and recall at various IoU levels and for objects of varying sizes. The results of the second copy, which had the highest result, were indicated.

- The F1-score curve illustrates the F1 score at different thresholds. An analysis of this curve can provide valuable information about the model's trade-off between incorrect positive predictions and incorrect negative predictions across various thresholds. *Figure 6* shows the F1-score curve for V2.

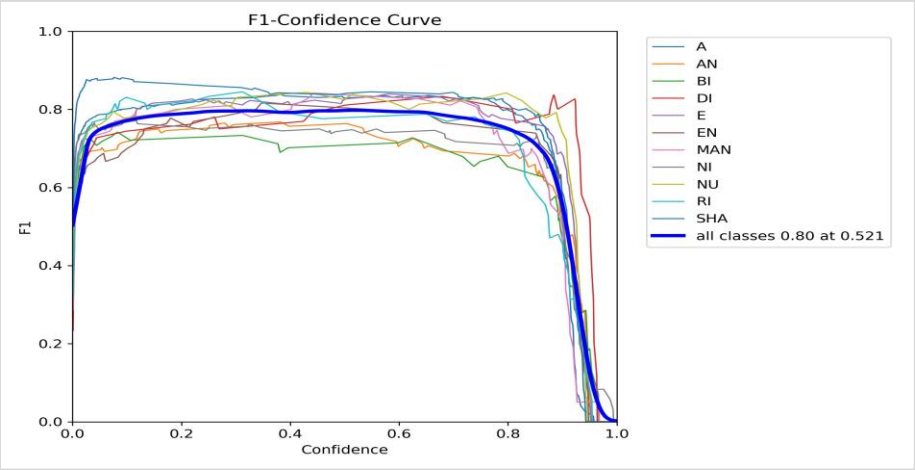


Figure 6 F1-score curve for V2

An average of 80% is the typical result for all object detection. However, an unbalanced F1-score indicates a disparity between precision and recall.

The precision curve demonstrates the precision values at various thresholds. Varying the threshold reveals how accuracy fluctuates along this curve. *Figure 7* depicts the precision – confidence curve, with the average result being 1.00 at a threshold of 1.000. As a crucial tool for visualizing classification problems, the precision-recall curve highlights the trade-offs between these two metrics at different thresholds, becoming particularly vital when dealing with unequal classes.

Figure 8 presents the precision-recall curve for V2, where the average results at mAP50 for all classes reach 0.827. *Figure 9* displays the recall curve for V2, showing a decrease in results as the threshold between the prediction and the ground truth increases. At a zero threshold, the average result for all classes is 0.92.

Figure 10 displays the confusion matrix for three versions of the split dataset. The prediction error rate, including FP and FN, frequently occurs in signals with a limited number of annotations. For instance, the number of annotations for each tag includes DI at 77 and the largest, tag A, at 1289. In the first version, BI had the highest percentage of incorrectly identified signals at 55%, with 45% identified correctly. In the second version, DI had the highest percentage of incorrectly identified signals at 75%, with correct identifications (true positives, TP) at 25%. In the third version, BI again had the highest percentage of incorrect signals at 55%, with TPs at 45%. Tag A recorded the highest percentage of correct identifications due to having the most

annotations, with TPs at 88%, 87%, and 88% for the respective versions. The FP and FN rates were 11%, 13%, and 11%, respectively.

YOLO object detection employs three loss functions as shown in *Figure 11*: box-loss (the regression value of the bounding box, MSE), obj-loss (the confidence of object losses), and cls-loss (the classification loss, cross-entropy). The validation box-loss, cls-loss, and df1-loss for the three versions were V1: 1.2078, 1.3489, 1.633; V2: 0.97018, 0.79408, 1.2563; and V3: 1.1075, 0.89497, 1.1982, respectively.

Modifications in hyperparameters or other factors can influence the model's performance, potentially increasing or decreasing the results.

Different learning rates, batch sizes, and other hyperparameters are employed to find a better ensemble and enhance performance. Occasionally, increasing the number of training epochs may not provide sufficient time for the model to learn the dataset's features. It is crucial that dataset annotations are accurate and properly formatted, as incorrect or misclassified data can detrimentally affect performance. In this paper, the best performance was achieved by maintaining the default parameters.

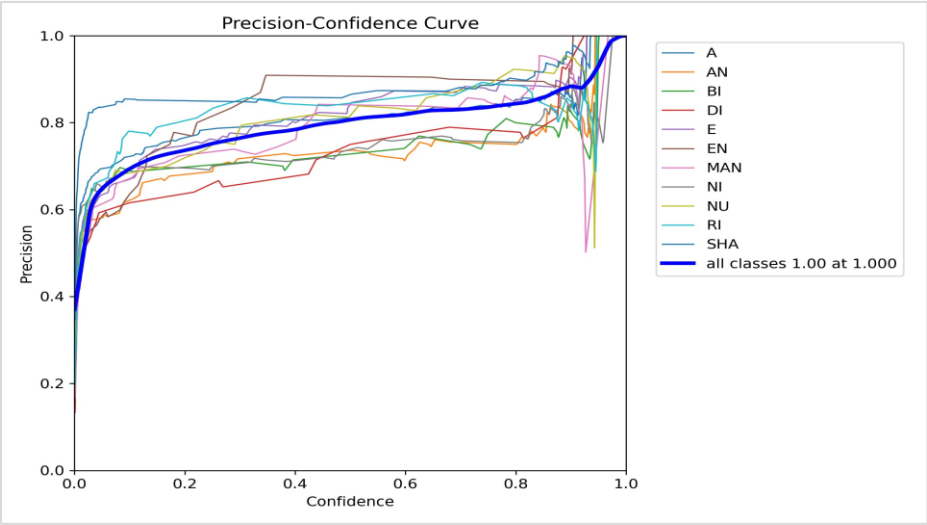


Figure 7 The precision - confidence curve for V2

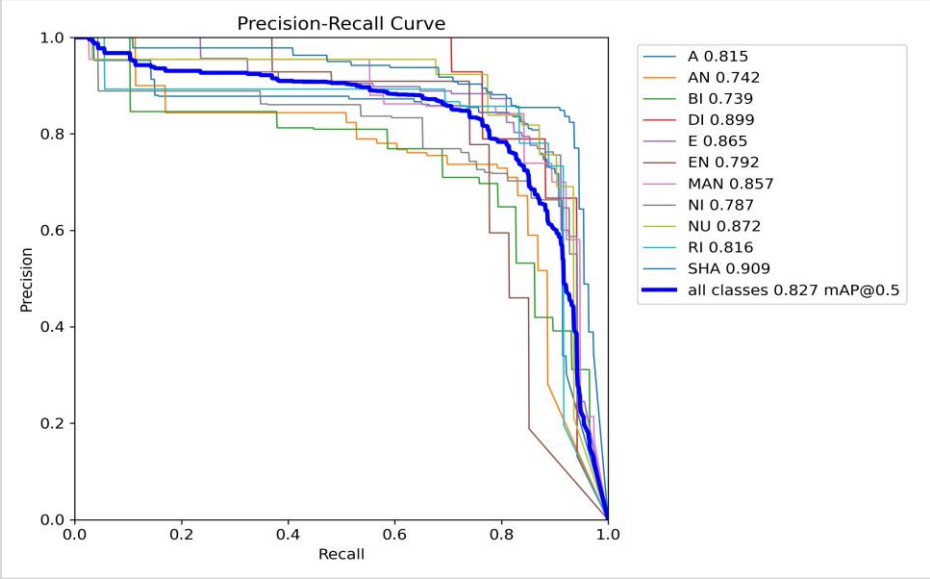


Figure 8 The precision-recall curve for V2

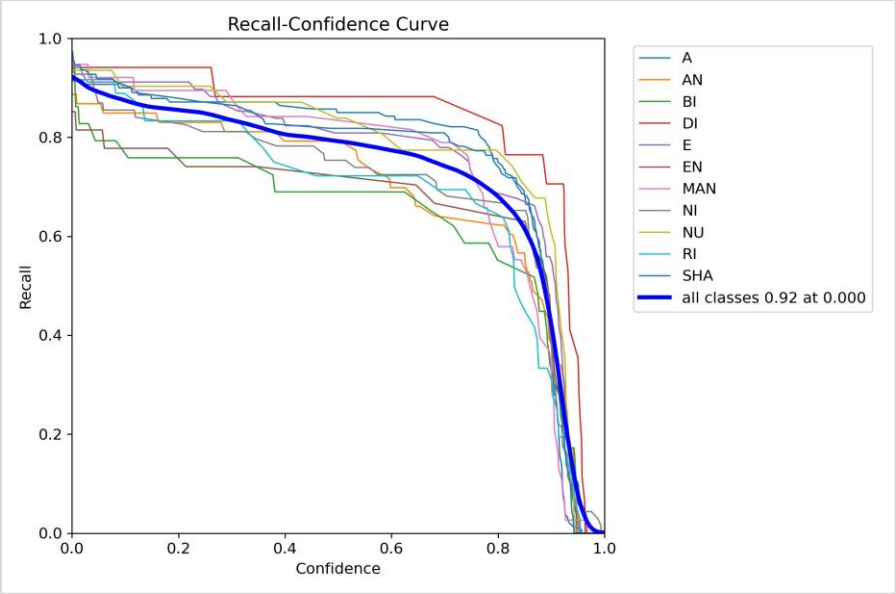
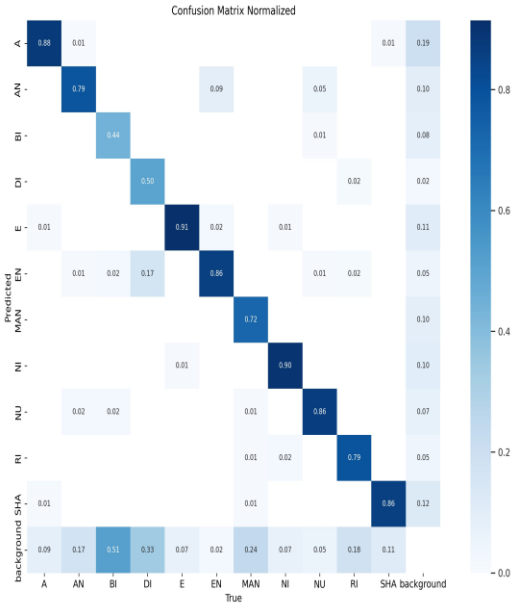
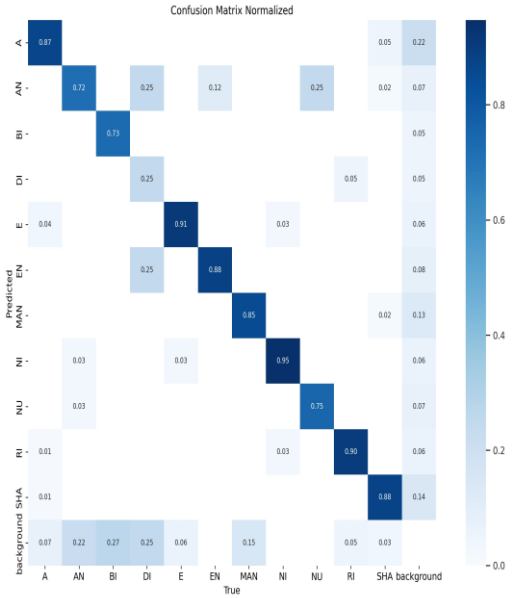


Figure 9 The recall-confidence curve for V2

Several frequently adjusted hyperparameters in Ultralytics YOLO are:

The learning rate is a parameter that controls the amount of the step taken at each iteration when approaching the minimum of the loss function. The batch size refers to the quantity of photos that are concurrently processed during a forward pass. An epoch refers to a single iteration where all the training instances are sent forward and backward through the model. Architecture details, such as the number of channels, levels, types of activation functions, etc.





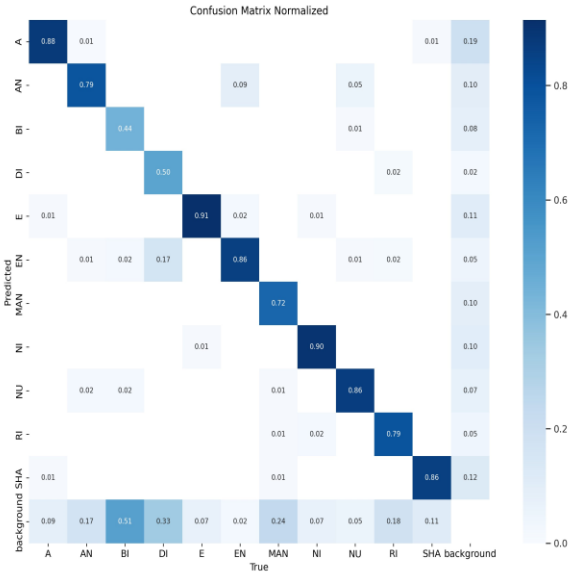
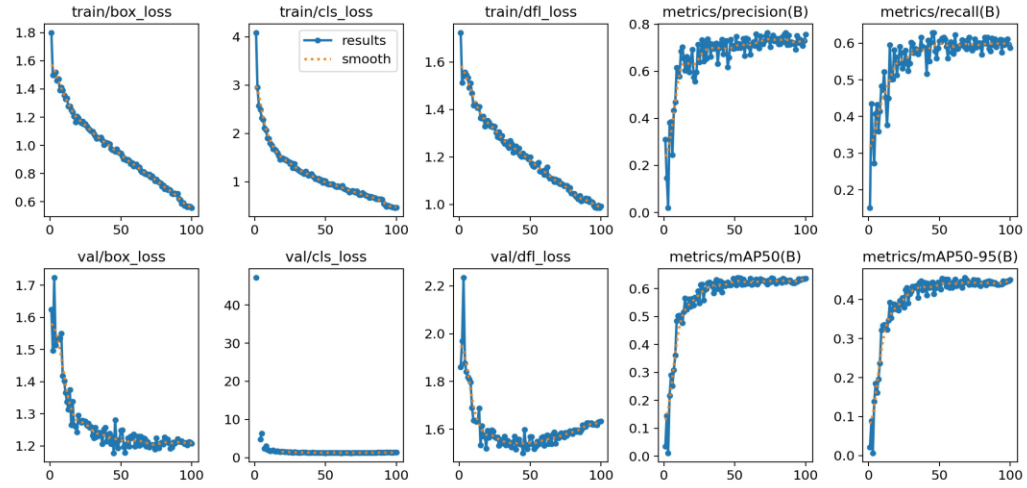
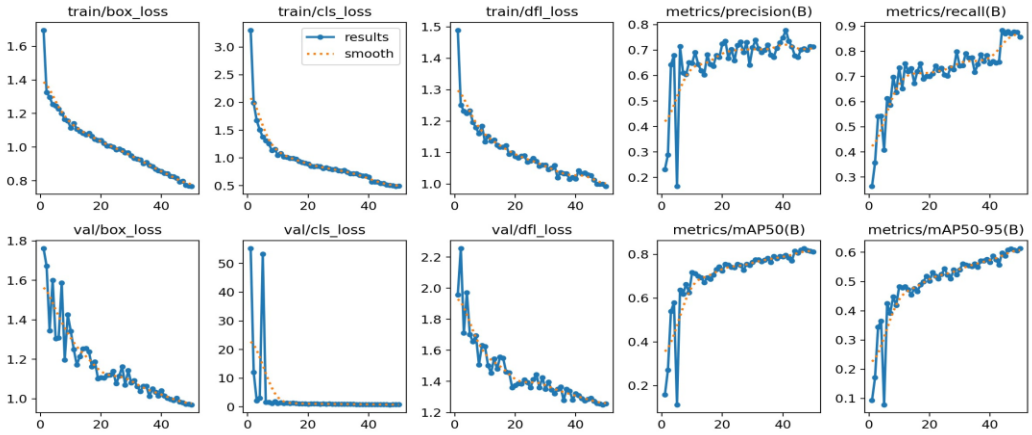


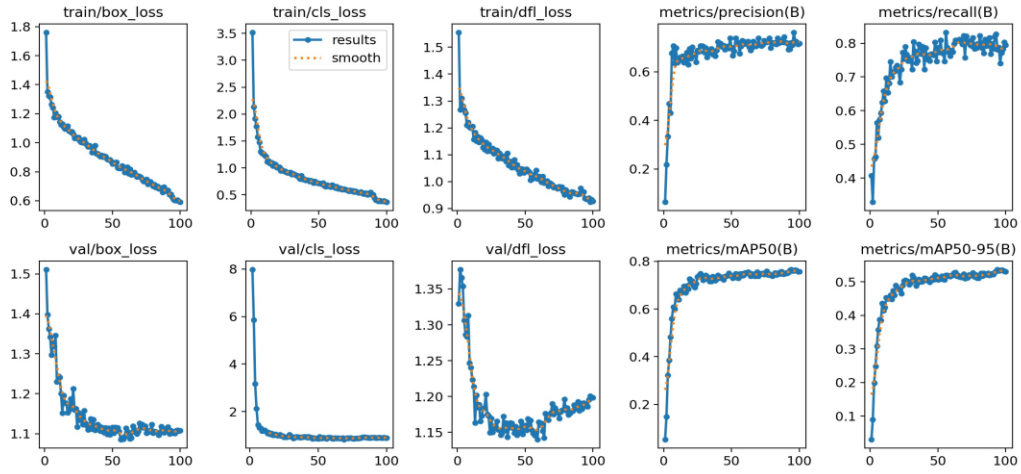
Figure 10 Confusion Matrix for three versions (V) of the split dataset



Version 1



Version 2

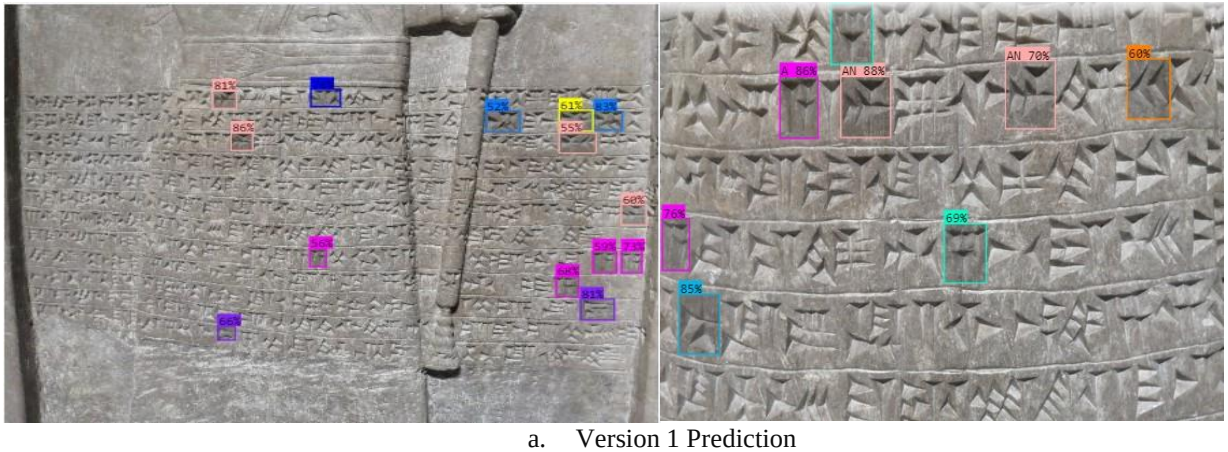


Version 3

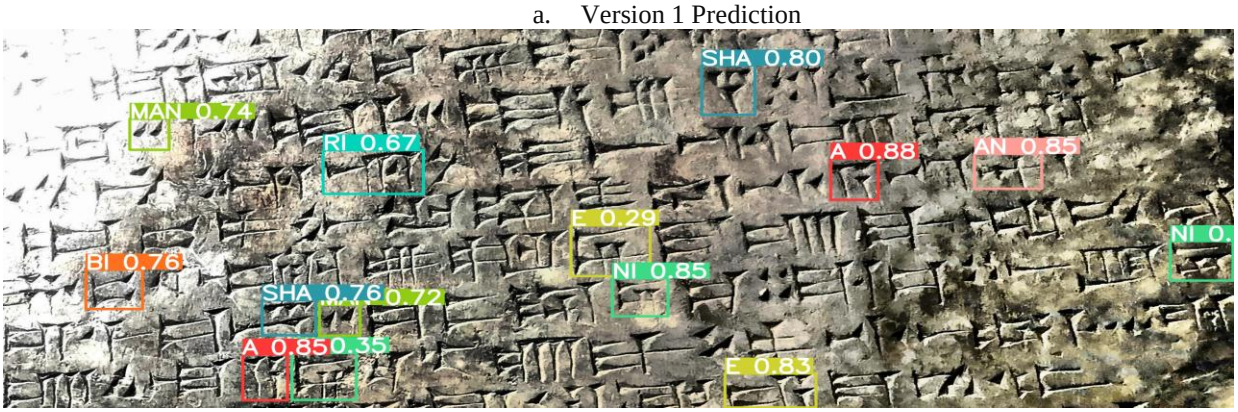
Figure 11 The Loss, mAP, precision, and recall results for three versions

4.2Prediction results

We see that the results are accurate whenever the image is high resolution and the signs are unmistakable. The accuracy begins to decline as the clarity of the image decreases, and the size of the signs becomes smaller. The image in *Figure 12* below is far away, and the signs are minor; after testing the V1, we see that not all signals detection but some of the signals are not detected. All results are excellent in the detection and labelling of the signals. The best result in version two is mAP50=82.7%.



a. Version 1 Prediction



b. Version 2 Prediction

c. Version 3 Prediction

Figure 12 a) V1 b) V2 and c) V3 prediction results

5.Discussion

Other research initiatives focusing on similar topics, specifically the identification of items from photographs using comparable frameworks, have reported notable results.

Ghosh et al. achieved a high accuracy rate of 96.46% in recognizing Bangladeshi signs using models developed on the MobileNet platform [51].

Hamplová et al. [14] attained a classification accuracy of 98.21% for 2000 Palmyrene letters per class by employing a custom convolutional neural network (CNN) architecture composed of 4 Convolutional/Max Pooling blocks. This study's contribution included compiling a dataset from the Iraqi Museum. Unlike previous research where images were sourced from the British Museum website or other sites without providing a dataset for a specific cuneiform language or dialect, this effort aimed at both collection and development for later use. The development utilized the latest algorithm from YOLO, specifically its eighth version created in 2023, which is regarded as one of the most accurate versions to date.

Some of the study implications are as under:

- The identification of the Assyrian signs was conducted with precision, achieving a mAP50 score of 82%. The identification process confirmed the presence of the Assyrian signs, which are consistent with the previously displayed images. The limited number of plates at the museum hindered the generation of robust data sets, resulting in a low level of precision in the obtained results. Furthermore, we have obtained highly favourable outcomes by employing the advanced yolov8x algorithm, which will be further enhanced in future research. Precision is the measure of the model's ability to detect true instances accurately, and it effectively reduces erroneous detections by 71%. The recall rate achieved a significant 86%, which is crucial for assessing the model's ability to recognize genuine objects accurately.
- These results are considered very good in terms of the number of images in the dataset and the explanatory signs for each Assyrian sign. The number of annotations for signs is unbalanced. Some signs are frequently repeated and have many annotations and other signs that are repeated infrequently and have few annotations. As for the number of images, it is also unbalanced, as mentioned previously. For each class, the minimum should be 1,500 images or more for the results to be accurate.
- The findings suggest that close-up photos were used to determine the tags, and clear tags were found to be more precise compared to distant and small tags. Some results showed identifications that closely resembled the original marks, either in appearance or due to erosion that made them similar to the desired mark.
- There are many limitations encountered, including unclear signs, damaged panels, some panels exposed to environmental conditions that led to

part of them being damaged, and the number of panels that are currently provided. There are many signs, up to 900 different categories of sign symbols with similar characteristics, making it difficult to distinguish them. Cuneiform writings often lack spaces between signs, and this also makes it difficult for us to identify the signs.

- To undertake and advance such research, it is imperative to gather unambiguous, high-resolution datasets and an ample quantity of photographs. We highly suggest further enhancing the outcomes by experimenting with alternative methods to identify any indicators that could potentially enhance the results using this particular dataset.
- Many studies have used other methods to identify signs, some of which use image processing by identifying the edges of signs. Remarkable outcomes were attained, and the symbols were recovered from ancient inscriptions. Other studies were conducted using other versions of YOLO and identifying the signs only (i.e., separating them from each other only) to facilitate reading the tablets. One of the studies was conducted using yolov5 achievable outcomes were attained, with a correct identification rate of 74%.

A complete list of abbreviations is shown in *Appendix I.*

6.Conclusion and future work

The YOLOv8 neural network architecture was designed to classify and locate strokes of Assyrian cuneiform signs in images of cuneiform tablets segmented into 640×640 pixels. The classifier based on YOLOv8x achieves an accuracy of 82.7%, a precision of 71.319%, and a recall of 85.647%. The success rate of YOLOv8x can be further increased by incorporating additional tagged images with a variety of characteristics, such as differing lighting, colors, and shadows, as well as by applying more advanced augmentation techniques.

In the future, efforts will be made to enhance accuracy by utilizing various neural network topologies, including region-based convolutional neural networks (RCNN) with selective search, which often achieve higher detection rates. Additionally, tuning of neural network topologies, including RCNN, will be initiated to attain the highest accuracy and facilitate better comparisons with newly labeled data.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

Author's contribution statement

Elaf A. Saeed and Munther A. Abdul Malik: Collaborated with an expert in cuneiform, to gather data and identify cuneiform symbols in order to initiate the training, Writing – review and editing. **Elaf A. Saeed and Ammar D. Jasim:** Collaborated with preparing the data set through labelling and training the model, Writing – review and editing.

References

[1] Rahma AM, Saeid AA, Hussien MJ. Recognize assyrian cuneiform characters by virtual dataset. In 6th international conference on information and communication technology and accessibility 2017 (pp. 1-7). IEEE.

[2] Abitbol R, Shimshoni I, Ben-dov J. Machine learning based assembly of fragments of ancient papyrus. Journal on Computing and Cultural Heritage. 2021; 14(3):1-21.

[3] Alstola T, Zaia S, Sahala A, Jauhiainen H, Svärd S, Lindén K. Aššur and his friends: a statistical analysis of neo-Assyrian texts. Journal of Cuneiform Studies. 2019; 71(1):159-80.

[4] Luo J, Hartmann F, Santus E, Barzilay R, Cao Y. Deciphering undersegmented ancient scripts using phonetic prior. Transactions of the Association for Computational Linguistics. 2021; 9:69-81.

[5] Sahala A. Contributions to computational Assyriology. Doctoral Dissertation, University of Helsinki. 2021.

[6] Snyder B, Barzilay R, Knight K. A statistical model for lost language decipherment. In proceedings of the 48th annual meeting of the association for computational linguistics 2010 (pp. 1048-1057). Association for Computational Linguistics.

[7] Fisseler D, Weichert F, Müller G, Cammarosano M. Towards an interactive and automated script feature analysis of 3D scanned cuneiform tablets. Scientific Computing and Cultural Heritage. 2013:16-7.

[8] Mara H, Krömker S. Vectorization of 3D-characters by integral invariant filtering of high-resolution triangular meshes. In 12th international conference on document analysis and recognition 2013 (pp. 62-6). IEEE.

[9] https://mimno.github.io/Mallet/index. Accessed 06 October 2023.

[10] Anderson SE, Levoy M. Unwrapping and visualizing cuneiform tablets. IEEE Computer Graphics and Applications. 2002; 22(6):82-8.

[11] https://github.com/ultralytics/ultralytics. Accessed 06 October 2023.

[12] Mara H, Krömker S, Jakob S, Breuckmann B. GigaMesh and gilgamesh: –3D multiscale integral invariant cuneiform character extraction. In proceedings of the 11th international conference on virtual reality, archaeology and cultural heritage 2010 (pp. 131-8). ACM.

[13] Hamdany AH, Omar-nima RR, Albak LH. Translating cuneiform symbols using artificial neural network. Telkomnika (Telecommunication Computing Electronics and Control). 2021; 19(2):438-43.
[14] Hamplová A, Franc D, Pavlíček J, Romach A, Gordin S. Cuneiform reading using computer vision algorithms. In proceedings of the 5th international conference on signal processing and machine learning 2022 (pp. 242-5).
[15] Rusakov E, Brandenbusch K, Fisseler D, Somel T, Fink GA, Weichert F, et al. Generating cuneiform signs with cycle-consistent adversarial networks. In proceedings of the 5th international workshop on historical document imaging and processing 2019 (pp. 19-24). ACM.
[16] Dencker T, Klinkisch P, Maul SM, Ommer B. Deep learning of cuneiform sign detection with weak supervision using transliteration alignment. Plos one. 2020; 15(12):1-21.
[17] Rusakov E, Somel T, Fink GA, Müller GG. Towards query-by-eXpression retrieval of cuneiform signs. In 17th international conference on frontiers in handwriting recognition 2020 (pp. 43-8). IEEE.
[18] Saeid AA, Rahma AM, Hussien MJ. Cuneiform tablets image preprocessing proposed algorithms techniques for pattern recognition. Iraqi Journal of Science. 2018:1326-38.
[19] Mara H. Multi-scale integral invariants for robust character extraction from irregular polygon mesh data (Doctoral dissertation). 2012.
[20] Howe NR. Part-structured inkball models for one-shot handwritten word spotting. In 12th international conference on document analysis and recognition 2013 (pp. 582-6). IEEE.
[21] Bogacz B, Gertz M, Mara H. Cuneiform character similarity using graph representations. 20th computer vision winter workshop 2015 (pp. 1-8).
[22] Bogacz B, Howe N, Mara H. Segmentation free spotting of cuneiform using part structured models. In 15th international conference on frontiers in handwriting recognition 2016 (pp. 301-6). IEEE.
[23] Massa J, Bogacz B, Krömker S, Mara H. Cuneiform detection in vectorized raster images. 21st Computer Vision Winter Workshop 2016 (pp. 1-9).
[24] Rothacker L, Fisseler D, Müller GG, Weichert F, Fink GA. Retrieving cuneiform structures in a segmentation-free word spotting framework. In proceedings of the 3rd international workshop on historical document imaging and processing 2015 (pp. 129-36). ACM.
[25] Rath TM, Manmatha R. Word spotting for historical documents. International Journal of Document Analysis and Recognition. 2007; 9:139-52.
[26] Rusiñol M, Aldavert D, Toledo R, Lladós J. Efficient segmentation-free keyword spotting in historical document collections. Pattern Recognition. 2015; 48(2):545-55.
[27] Saeid AA, Rahma AM. Cuneiform symbols recognition by support vector machine (SVM). Journal of AL-Qadisiyah for Computer Science and Mathematics. 2019; 11(1).
[28] Sahala A, Silfverberg M, Arppe A, Lindén K. Automated phonological transcription of Akkadian cuneiform text. In proceedings of the 12th conference on language resources and evaluation 2020. European Language Resources Association (ELRA).
[29] Liu L, Ouyang W, Wang X, Fieguth P, Chen J, Liu X, et al. Deep learning for generic object detection: a survey. International Journal of Computer Vision. 2020; 128:261-318.
[30] Everingham M, Van GL, Williams CK, Winn J, Zisserman A. The pascal visual object classes (VOC) challenge. International Journal of Computer Vision. 2010; 88:303-38.
[31] Ren S, He K, Girshick R, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. Advances in Neural Information Processing Systems. 2015.
[32] Redmon J, Farhadi A. Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767. 2018.
[33] Diwan T, Anirudh G, Tembhurne JV. Object detection using YOLO: challenges, architectural successors, datasets and applications. Multimedia Tools and Applications. 2023; 82(6):9243-75.
[34] Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S. Feature pyramid networks for object detection. In proceedings of the conference on computer vision and pattern recognition 2017 (pp. 2117-25). IEEE.
[35] https://roboflow.com/model/yolov8. Accessed 06 October 2023.
[36] Jiao L, Zhang F, Liu F, Yang S, Li L, Feng Z, et al. A survey of deep learning-based object detection. IEEE Access. 2019; 7:128837-68.
[37] Jiang P, Ergu D, Liu F, Cai Y, Ma B. A review of Yolo algorithm developments. Procedia Computer Science. 2022; 199:1066-73.
[38] Li Z, Peng C, Yu G, Zhang X, Deng Y, Sun J. Detnet: a backbone network for object detection. arXiv preprint arXiv:1804.06215. 2018.
[39] Zheng Z, Wang P, Liu W, Li J, Ye R, Ren D. Distance-IoU loss: faster and better learning for bounding box regression. In proceedings of the AAAI conference on artificial intelligence 2020 (pp. 12993-3000).
[40] Li X, Wang W, Wu L, Chen S, Hu X, Li J, et al. Generalized focal loss: learning qualified and distributed bounding boxes for dense object detection. Advances in Neural Information Processing Systems. 2020; 33:21002-12.
[41] Terven J, Cordova-esparza D. A comprehensive review of YOLO: from YOLOv1 and beyond. arXiv 2023. arXiv preprint arXiv:2304.00501. 2023.
[42] Tamang S, Sen B, Pradhan A, Sharma K, Singh VK. Enhancing covid-19 safety: exploring yolov8 object detection for accurate face mask classification. International Journal of Intelligent Systems and Applications in Engineering. 2023; 11(2):892-7.

[43] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft coco: common objects in context. In computer vision–ECCV 2014: 13th European conference, Zurich, Switzerland, 2014 (pp. 740-55). Springer International Publishing.
[44] Lee Y, Kim T, Lee SY. Voice imitating text-to-speech neural networks. arXiv preprint arXiv:1806.00927. 2018.
[45] Gordin S, Gutherz G, Elazary A, Romach A, Jiménez E, Berant J, et al. Reading Akkadian cuneiform using natural language processing. PloS one. 2020; 15(10):1-16.
[46] Gutherz G, Gordin S, Sáenz L, Levy O, Berant J. Translating Akkadian to English with neural machine translation. PNAS nexus. 2023; 2(5):1-10.
[47] Jauhiainen T, Jauhiainen H, Alstola T, Lindén K. Language and dialect identification of cuneiform texts. arXiv preprint arXiv:1903.01891. 2019.
[48] Cho J, Lee K, Shin E, Choy G, Do S. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? arXiv preprint arXiv:1511.06348. 2015.
[49] Huang Z, Li L, Krizek GC, Sun L. Research on traffic sign detection based on improved YOLOv8. Journal of Computer and Communications. 2023; 11(7):226-32.
[50] Ahmad T, Ma Y, Yahya M, Ahmad B, Nazir S, Haq AU. Object detection through modified YOLO neural network. Scientific Programming. 2020; 2020:1-10.
[51] Ghosh T, Abedin MM, Chowdhury SM, Tasnim Z, Karim T, Reza SS, et al. Bangla handwritten character recognition using MobileNet V1 architecture. Bulletin of Electrical Engineering and Informatics. 2020; 9(6):2547-54.

Elaf A. Saeed serves as a systems and control engineer at the University of Al-Nahrain, College of Information Engineering, in Iraq. Her expertise spans control, Embedded Systems, Artificial Intelligence, IoT, and Web Design. Recognized for her programming talent, Elaf has authored eleven books published by Lambert Academic. Distinguished as the top student throughout her B.Sc. studies, she also brings four years of teaching experience to her role. Currently, she is pursuing her research as a master's student.
Email: elafe1888@gmail.com

Dr. Ammar D. Jasim is an Assistant Professor at the College of Information Engineering, Al-Nahrain University, Iraq. He earned a Ph.D. in information engineering with a specialization in networks. His research interests encompass Networks, Communications, Security, and Artificial Intelligence. Currently, he leads the Information System Department at Al-Nahrain University, Iraq. Additionally, he supervises several Ph.D. students in their research endeavors.
Email: ammar.alaythawy@nahrainuniv.edu.iq

Dr. Munther A. Abdul Malik is affiliated with the University of Baghdad, College of Arts, specializing in Archaeological and Assyriology studies. His expertise lies in reading, translating, and transliterating cuneiform texts. His research focuses on Sumerian, Akkadian, Babylonian, and Assyrian civilizations, and he is proficient in translating texts from all these ancient cultures. Munther actively participates in excavations at Sippar and Assur and has completed numerous training courses in archaeology and modern technical surveys.
Email: munther@coart.uobaghdad.edu.iq

Appendix I

S. No.	Abbreviation	Description
1	2D	Two-Dimensional
2	3D	Three-Dimensional
3	ANN	Artificial Neural Network
4	API	Application Programming Interface
5	AP	Average Precision
6	BCE	Before the common era
7	BoF	Bag-of-Features
8	CIOU	Complete Intersection Over Union
9	CLI	Command Line Interface
10	CNN	Convolutional Neural Network
11	COCO	Common Objects in Context
12	CSP	Cross Stage Partial
13	CSPLayer	Cross-Stage Partial Layer
14	CUDA	Compute Unified Device Architecture
15	DFL	Distribution Focal Loss
16	FPS	Frames Per Second
17	GPU	Graphical User Interface
18	gTTS	Google Text-to-Speech
19	HMMs	Hidden Markov Models
20	IDs	Sign Identifications
21	IOU	Intersection Over Union
22	mAP	Mean Average Precision
23	MLP	Multi-Layer Perceptron
24	MSE	Mean Square Error
25	MS COCO	Microsoft Common Objects in Context
26	QbX	Query-by-Expression
27	OCR	Optical Character Recognition
28	PIP	Package Manager for Python
29	RCNN	Region-Based Convolutional Neural Networks
30	YOLO	You Only Look Once
31	YOLOV8	You Only Look Once Version 8