A Neural Framework For Handwritten Calendar Parsing and Semantic Content Categorization

Antoni Gagliard[1], Rayappa David Amar Raj[1] and Rama Muni Reddy Yanamala[2]

[1]*National Institute of Technology, Warangal, Telangana, 506004, India*
[2]*Amrita School of Artificial Intelligence, Amrita Vishwa Vidyapeetham, Coimbatore, Tamil Nadu, 641112, India*

Abstract
Digital calendars, accessible via laptops, tablets, and smartphones, offer features such as automatic reminders that improve time management and personal organization. However, older people often struggle to use these tools, preferring to rely on traditional paper calendars. This digital divide can lead to missed appointments and a subsequent negative impact on well-being. We propose an innovative application that can automatically capture and digitize a physical calendar, allowing reminders to be sent and commitments to be tracked even by third parties. By integrating the familiar interface of paper with digital features, our tool aims to improve appointment keeping and reduce the technological gap in time management for the elderly population.

Keywords
Artificial Intelligence, Machine Learning, Deep Learning, Optical Character Recognition

1. Introduction

Optical Character Recognition (OCR) refers to a set of techniques used to detect and convert characters from physical documents into editable, and searchable digital text. This process typically involves capturing an image of the document using a scanner or a digital camera. The ability of converting various forms of documents can be applied in a wide range of fields, for example the recognition of human handwriting, the digital conversion of labels and manuscripts, the recognition of numerical digits in financial and banking contexts and the validation of a particular type of handwriting to authenticate the provenance of a manuscript. Another interesting application that has been developed in recent years involves the recognition of ancient characters [1]. The objective of the challenge was to digital reconstruct ancient damaged papyrus scrolls. The scrolls were digitally "unwrapped" using computed tomography (CT) and machine-learning technology. The resulting scans were then turned into a 3D volume of voxels, which have been segmented by tracing the crumpled layers of the rolled papyrus in the 3D scan, actually flattening the images. The last step was detecting ink on papyrus by using machine learning to identify regions of ink in the flattened segments of the papyrus. A particularly remarkable aspect of this application is that the model operated without any prior knowledge of alphabets or handwriting conventions. The digital characters predicted by the model, result therefore purely from plotting the local ink detection spots across a generated image. Another important application of handwriting recognition has been applied to cuneiform tablets [2]. Researchers, instead of using photos, relies on 3D models of the tablets, delivering significantly more reliable results than previous methods. This makes it possible to search through the content of multiple tablets and to compare them with each other. They used 3D models of nearly 2000 cuneiform tablets, many of them are more than 5000 years old and are thus among mankind's oldest surviving written records. What they discover is an extremely wide range of topics, from shopping lists to court rulings, providing a glimpse into mankind's past several millennia ago. However, despite the result obtained, the challenge remains open since there are lot of complications, mostly the fact that some tablets are heavily ruined and also that the writing system was very complex at that age and encompassed several languages. Consequently, effective modeling requires not only higher-quality data but also more sophisticated prior knowledge [3, 4] to capture the complexity and multilingual nature of cuneiform writing.

It is noteworthy that also a major technology company such as Google has developed its own personal OCR system in the more recent years. Google OCR, developed by Google AI [5], is designed to convert a variety of document types, including scanned documents, PDFs, and images captured by a digital camera, into editable text. The system's principal advantages are its high degree of accuracy, achieved through the use of sophisticated deep learning techniques for the recognition and extraction of text with remarkable precision (even in the presence of complex backgrounds or low-quality images), the incorporation of multiple languages within the system leads to the capability of processing a wide range of alphabets,

ICYRIME 2025: 10th International Conference of Yearly Reports on Informatics, Mathematics, and Engineering. Czestochowa, January 14-16, 2025
yrm\protect1_reddy@cb.amrita.edu (R. R. Yanamala)

including ideograms, and possibility to process not only printed characters but also handwritten texts.

OCR system plays also an important social role when it is employed in all that applications to deal with visual impairments, giving the possibility to blind people to convert a written text into audio (OCR + speech synthesis). In conclusion, OCR represents a versatile technological solution with broad applicability across document processing, data management, and accessibility domains. In order to achieve our scope, it is necessary to create a pipeline that can be employed to digitize the content, assign it to a category and store it in a database or in an existing digital calendar, for example using the Google Calendar API or the iOS Calendar API.

Challenges The process of recognizing and digitizing human handwriting presents several significant challenges for OCR systems. The presence of noise and distortions in images represents a considerable obstacle, as it can negatively impact the efficiency and accuracy of the system. They may also struggle to recognize characters in scanned images affected by distortions or intrinsic noise, leading to recognition errors. Furthermore, the issue of multilingual support introduces another layer of complexity, as OCR systems may face challenges in processing documents that contain multiple languages, each with its own set of characters and linguistic rules. OCR systems, like human readers, are inherently tied to specific alphabets when recognizing characters. The system learns the local features of the different characters directly from the handwritten text, there are input data in the form of $I = (x_i, y_i)$, $y_i = \{y_{(i,1)}, y_{(i,2)}, \dots, y_{(i,N)}\}$, where each $y_{(i,j)}$ is a symbol, or **grapheme** (from gr. $\gamma\rho\alpha\phi\omega$, 'write'), mapped in a digital encoding and decoding alphabet. The system learns a specific alphabet directly from the text and since local features (shape, thickness, corners, edges,...) are inherent to the characters of an alphabet, As a result, the system experiences a significant drop in performance when applied to characters outside its trained alphabet. This significantly limits the **transfer learning techniques**, since the network must be retrained from scratch to recognize another alphabet different from the one on which the model has already been trained.

Moreover, the diversity of handwriting styles further complicates the OCR process. Handwriting is highly individual, requiring OCR systems to handle not only personal variation but also atypical styles such as cursive or artistic fonts. The inherent subjectivity of handwriting makes it imperative to develop an OCR system that is robust: it should handle as much as possible the actual changes in font and style from text to text and decipher the myriad ways in which individuals express themselves on paper. Taking the Latin alphabet into account, we can differentiate the characters into **capital letters** and **lowercase letters**. Notably, capital letters tend to exhibit lower variability, whereas lowercase letters—though subject to basic calligraphic conventions—reflect more personal handwriting traits and a broader range of stylistic variation.

2. Related Works

Early efforts in text digitization date back to the development of LeNet [6], which demonstrated that a shallow convolutional neural network could accurately recognize handwritten digits in 32×32 grayscale images. Building on this, [7] proposed a methodology leveraging the MNIST dataset [8] to address more complex handwriting recognition tasks, but also coping with image defects and noise [9]. Their approach emphasized preprocessing steps—including grayscale normalization, cropping, and resizing—to improve the recognition of isolated handwritten characters. They showed the effectiveness of convolutional neural networks (CNNs) in extracting local features from such inputs. The utility of CNNs for handwriting recognition has since been widely adopted. In 2015, [10] introduced the CRNN architecture, which combines CNNs for spatial feature extraction with recurrent neural networks (RNNs) to model character sequences. This design is particularly suited for handwritten word recognition, as RNNs can capture sequential dependencies—albeit with limitations such as the vanishing gradient problem. More recently, [11] proposed a system that combines CNNs with Error Correcting Output Codes (ECOC) to enhance classification robustness. Feature extraction is performed using architectures such as LeNet [6] and AlexNet [12], while classification is carried out by training an ensemble of binary Support Vector Machines (SVMs) via ECOC. This method decomposes the multiclass problem into several binary subproblems, yielding higher accuracy compared to CNNs followed by a standard softmax classifier—particularly on the MNIST dataset. In 2021, [13] introduced a more complex model based on CRNNs for full handwritten document recognition. This approach integrates CNNs for visual feature extraction with Long Short-Term Memory (LSTM) networks to model sequential dependencies across words or phrases. The system is trained using the Connectionist Temporal Classification (CTC) loss function [14], which enables sequence prediction without requiring explicit character-level alignment. CTC considers all possible alignments and computes a summed probability, allowing for end-to-end training even when segmentation is ambiguous. The next leap in this domain has been driven by the application of transformers [15] to computer vision, particularly through the introduction of the Vision Transformer (ViT) [16]. ViT replaces convolutional layers with a pure attention mechanism, enabling the model to capture long-range dependencies in visual inputs. This shift opens new directions in handwriting recognition by enabling the integration of global context across entire input images, beyond the local receptive fields of traditional CNNs.

It is worth pointing out that, before the advent of CRNN and ViT model, the majority of the proposed achieved high performance in terms of recognition accuracy but showed huge limitations: the input for the first OCR neural model was necessarily provided in the form of individual character alphabet, the networks were able to classify the salient features of the character and provide a classification consisting of the corresponding digital label, but with the inability to create a context (both in terms of previous word in a sentence and in terms of individual characters in a single word), profound refactoring of the software systems was often required[17, 18].

3. Implementation Description

To ensure accurate calendar digitization, the proposed pipeline is composed of several neural modules. The initial component is a **segmentation network** based on the U-Net architecture [19], which enables the model to identify the most semantically relevant regions of the calendar. This network produces a segmentation mask, allowing for the algorithmic extraction of targeted calendar segments. Once the relevant segments are extracted, they are processed by two separate ResNet6-based convolutional neural networks [20], responsible for recognizing the month and day digits, respectively. While the recognition of day digits benefits from standardized patterns—since the digits recur uniformly across samples—the month field presents greater variability. Users are allowed to write the month manually, which introduces inconsistencies in handwriting style and positioning. To address these variations and improve classification robustness, the calendar template incorporates a distinct pair of AprilTags [21] for each month of each year, positioned at the left and right margins of the month field. This design introduces spatial regularity, guiding the convolutional layers toward more stable visual features and mitigating the ambiguity caused by user-written titles. The final step of the pipeline involves processing the reminder segments, where a second U-Net is employed to segment user-written sentences into individual words. These word segments are then passed to a digitization module that combines the Vision Transformer (ViT) [16] and BERT [22]. While ViT captures visual features at a global scale, BERT processes the embedded representations to extract semantic meaning. An additional component of the system enables content classification at the line level. Specifically, each textual line is processed through BERT for semantic categorization. With minimal additional effort, this classification can be extended to the entire calendar: feature vectors from each line are aggregated and averaged to obtain a single representation, which is then used to predict a high-level category for the calendar's overall content.

Dataset	Training	Validation
Calendar	146	74
Numeric Digits	682	527
IAM Lines/Sentences	7561	3781
IAM Words	64302	32152

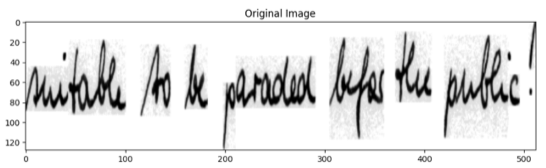
Table 1
Datasets: the calendar dataset has been created manually by using a graphic tablet, numeric digits is a collections of numbers in the range $[1, 31]$ generated from computer system fonts, while IAM (Institut für Informatik und Angewandte Mathematik) Handwritten is a dataset belonging to the University of Bern.

3.1. Dataset

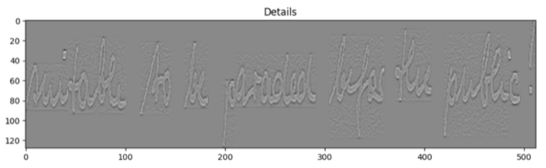
To conduct the experiments, we independently created a dataset of handwritten calendar images using a graphic tablet. The corresponding ground truth segmentation masks were annotated using the Oxford VGG Image Annotator tool. The dataset, although limited in size (see Table 1), is sufficient for our use case. Unlike generic segmentation tasks—such as those involving the heterogeneous images found in datasets like ImageNet—our domain involves structurally homogeneous data. All input images depict the same subject: a calendar with a symmetric and well-defined layout. This structural regularity allows for meaningful learning even with a smaller number of examples. To simulate realistic acquisition conditions, we applied synthetic distortions using Photoshop to introduce parallax effects (horizontal, vertical, or both). These transformations reflect the common scenario where the camera capturing the calendar may not be perfectly aligned with the sheet. Incorporating such distortions during training improves model robustness in real-world scenarios. To correct these geometric distortions at inference time, we integrated a **RANSAC** (Random Sample Consensus) [23] module into our pipeline. The four AprilTags placed at the corners of each calendar template serve as keypoints, allowing RANSAC to estimate a homography and realign the captured image with the reference calendar layout.

The numeric digits dataset is assembled by saving pictures of numbers from 1 to 31 using classical fonts available at the level of the computer's operating system. This choice is useful both for training a compact ResNet to recognize calendar day numbers and for allowing the model to generalize beyond the specific font used in the study.

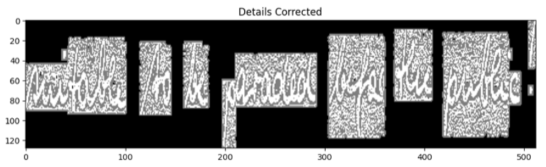
Other useful data are retrieved from the public Internet. In particular, the IAM Handwriting Datasets [24] are used to make the pre-trained of the Word Paser model and of the Note Digitizer. IAM Handwriting is a dataset that contains a collection of words cut from portions of written texts, nd its main advantage, as discussed in the introduction, lies in enabling the model to learn relevant character features. This approach enables the system to learn the sequential features of the characters that together make up a particular word.



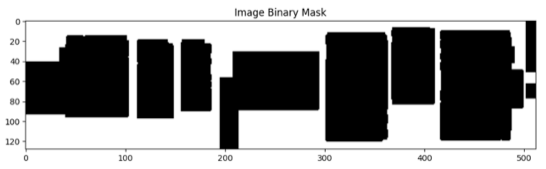
(a) Example of the original image taken from IAM Lines dataset, "suitable to be paraded before the public"



(b) The original image is filtered with the Laplacian operator to retrieve its details, in particular the edges



(c) Pixels values of the detail are corrected in order to be 0 128 255. The filtering is applied taking into account some intensity values as threshold.



(d) Final binary inverse thresholding

Figure 1: IAM Lines Mask Creation: since each sentence contained in the IAM lines dataset is constructed as a collage of words cuttings, there's a gray scale level provided that denote the break between the white background and the "internal" background of the little piece of word. This information can be exploited to construct more or less accurate binary word masks to trained a second **U-Net** to learn the text line segmentation.(1d)

3.2. Data Processing

Standard computer vision techniques are employed to process the images contained in the datasets: Gaussian blurring is preferred over classical blurring, since using a kernel derived from a Gaussian distribution helps to attenuate image noise and details, ensuring better preservation of edges and contours. The image pixel intensity values are uniformed by applying a binary thresholding technique that sets to a maximum value all pixels above a certain threshold (represented, for example, in the case of the IAM dataset, by the grayscale levels annotated in the data), while setting the values below to 0. Moreover, all pixel values in the image are additionally scaled by a factor of $1/255$ to bring them into the interval $[\,0, 1\,]$, and normalized around a certain mean and standard deviation. These general steps ensure that data are in a format more suitable for model processing.

Regarding the calendar dataset, to ease the computational segmentation process, images are downsampled using a Laplacian Pyramid. The reason why this technique is applied to the calendar data before feeding them into the network will be made clearer in Section 3.4.1.

It is also worth noting that, since a word segmentation block for the calendar note segments has been included in the architecture, it needs to be trained on pairs (image, word mask) that are not directly available in the dataset. This is why a dedicated image processing step is applied online during dataset generation to obtain, from a single text line image, its corresponding ground-truth word segmentation mask. The original image (Figure 1a) is filtered with a Laplacian operator in order to extract its details, borders, and edges (Figure 1b). This operation is particularly useful in the current data domain because, as previously mentioned in Section 3.1, the images from the IAM Line/Sentence datasets are collections of cropped word/text lines. By further filtering the pixel values of the detailed image, it becomes possible to highlight borders and edges, and thus delineate the cutting boundaries with respect to their content (Figure 1c). At this point, by applying a binary inverse threshold function, the final word-level binary segmentation mask can be obtained (Figure 1d).

3.3. Neural blocks

The digitization system is composed of five neural modules working in parallel. The first is the Calendar Parser, a U-Net model with encoder-decoder structure and skip connections, which takes as input a calendar image (downsampled using a Laplacian Pyramid) and performs pixel-wise classification into six semantic regions. Initially, a larger number of classes was tested to directly model the month and day digits at this stage, but this was later simplified, as discussed in Section 3.4. The

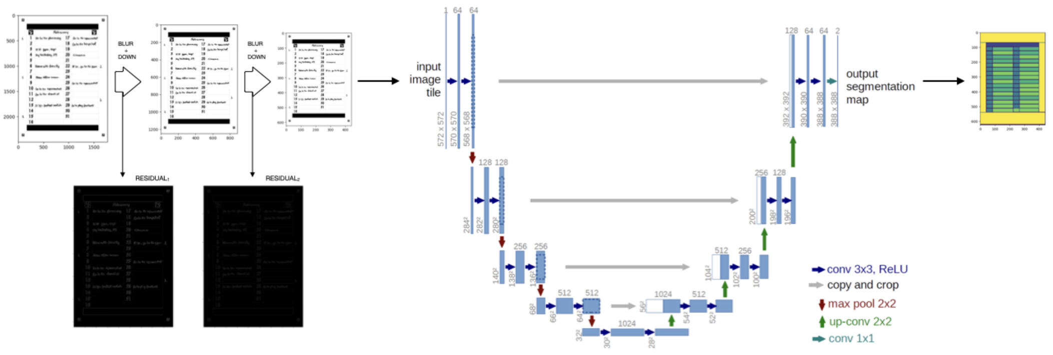


Figure 2: Calendar Parser: it represents the first neural block of the overall digitization system. The model, represented by a **U-Net** network takes as input a downsample image of a calendar and predict as output its segmentation map. It worthy to note that the input image is scaled by using a Gaussian and Laplacian Pyramids both for two reason: 1) providing an input that is not too much big for the model in order to help its computations. 2) Once the calendar fragment are retrieved thank to segmentation map is important to bring them back to the original sizes by adding their corresponding residuals stored from the pyramid downsampling process. This ensure to have images that are poor in resolution for the next digitization steps.

Word Parser is also based on a U-Net architecture and is pre-trained using image/mask pairs from the IAM Line/Sentence dataset, where masks are generated as described in Section 3.2. It produces a binary segmentation mask for each text line, and, as shown in Section 4, generalizes well to calendar data despite significant visual differences from the training set. Both the Month Digitizer and Day Digitizer are implemented as compact ResNet architectures composed of six convolutional layers with two residual connections. Each model concludes with a linear classification head, consisting of either 12 or 31 output units depending on whether the task is month or day classification, respectively. This minimal design was chosen to ensure fast inference while maintaining adequate performance given the low visual variability of the segmented inputs.

3.3.1. Content classification with BERT

Once the handwritten text has been digitized, it may be useful to classify it into categories to highlight the nature of its content. To do so, it is possible to rely on a Transformer-based architecture modeled by a modern LLM such as BERT. BERT is a Transformer architecture consisting of a single encoder. Its peculiarity lies in the fact that, given the nature of its training (Next Sentence Prediction and Question Answering tasks), it can be easily adapted to a new domain by fine-tuning the encoder on the target dataset and stacking an additional classification layer on top of its head. BERT incorporates a strong semantic understanding of the English language and can therefore be used to process textual content.

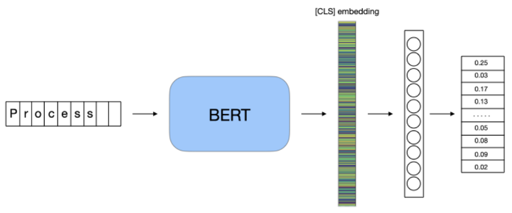


Figure 3: Text Classification Component: relying on BERT LLM architecture it is mainly composed of a fully connected that act as a classifier of `[CLS]` special token features. `CLS` BERT special token embed the overall meaning of the entire sentence resulting very suitable to classify the text's content.

Content Text Classifier To classify the textual content, each sentence is preprocessed by lowercasing all words and removing stopwords to reduce noise and retain only the most relevant terms. The processed text is then passed through BERT, which maps each token to a 768-dimensional latent space. The special `[CLS]` token, capturing the overall sentence meaning, is extracted and fed into a linear classification layer. A Softmax function is applied to the output to obtain a probability distribution over predefined content categories. The model is trained using multiclass cross-entropy loss.

3.4. Experiment

The overall project is made up of a bunch of experiments that have been conducted on the different architectural components examined in Section. 3.3: since both the calendar parser and the word parser take benefit of using U-Net segmentation architecture, both the model has been trained using **Dice Loss** [25]. The Dice-Sørensen coefficient, Equation. 1, also know as Dice Coefficient, is a statistic used to gauge the similarity of two samples. It is widely employed in the field of segmentation, where it is particularly effective in evaluating the pixel-wise accuracy of ground truth masks in comparison to the output segmentation mask generated by a model.

$$Dice(\ P, G)\ = \frac{2|P \cap G|}{|P| + |G|} \tag{1}$$

P and G are the set of pixels belonging respectively to the predicted and the ground truth masks, $|P \cap G|$ is the number of pixels that both set have in common and finally $|P|$ and $|G|$ represent the total amount of pixel in the two sets. The calendar parsing is essentially a `MultiClassification` task. The Equation. 1 becomes:

$$Dice(\ P, G)\ = \frac{1}{C}\sum_{c=1}^{C}\left(1 - \frac{2\sum_{n=1}^{N}(\ P_{i,c} \cdot G_{i,c})}{\sum_{n=1}^{N} P_{i,c} + \sum_{n=1}^{N} G_{i,c}}\right) \tag{2}$$

the word parsing can be treated instead as a `BinaryClassification` task and the Equation. 1 becomes:

$$Dice(\ P, G)\ = \frac{2\sum_{n=1}^{N}(\ P_i \cdot G_i)}{\sum_{n=1}^{N} P_i + \sum_{n=1}^{N} G_i} \tag{3}$$

P_i is the predicted binary value for pixel i and could be $P_i = 1$ or $P_i = 0$, G_i is the same but in the ground truth mask while N is the total number of pixel, and since the task deals with 2D images, $N = W \times H$, where W and H are respectively the width and the height of the images. The value of the Dice Loss can be easily obtained as:

$$Dice_{Loss}(\ P, G)\ = 1 - Dice(\ P, G) \tag{4}$$

Looking at the Equation. 4 it can observed that minimize the loss means effectively maximize the Dice Coefficient that provides a measurement of the similarity between the model's prediction and the ground truth. Once the model is trained, the most suitable prediction of the segmentation map is obtained respectively as $P^* = argmax_c(\ P_{i,c})$ in a multi-class domain and in the simpler binary case as $P^* = \mathbf{1}(\ P_i \geq 0.5)$, with $\mathbf{1}$ that is an indicator of whether the condition is satisfied or not.

3.4.1. Design Choices

The images are converted from 3 channels (RGB) to 1 channel (black and white) and processed as described in Section 3.2. Calendar images are downsampled using a Laplacian Pyramid before being fed into the network. This technique is particularly useful, as each blurring-downsampling step retains the image's residuals. Once the calendar is segmented, each relevant segment can be upsampled and added back to its corresponding residual in the pyramid to restore the original resolution. This design choice is especially effective in addressing the problem of image resolution, which would otherwise be too low for proper digitization.

Regarding resizing operations, day segments are scaled to 32×32, while month fragments are resized to 128×512. It is also worth noting that since a ViT [16] is used in the note digitization phase, word images from the IAM Words dataset are resized to 224×224. This dimension allows the Vision Transformer to correctly apply its initial convolutional step, which extracts patches using a 16×16 kernel with a stride of 16×16.

One final note concerns the `VisualEncoderDecoder` model: `HuggingFace` provides a wrapper class combining a vision transformer with a language model decoder. Instead of using the pre-trained **TrOCR** [26], a new model instance was created, modifying the first convolutional layer of ViT to accommodate the specific input conditions of this experiment. This adjustment was necessary because ViT was originally trained on ImageNet [27], where input images have three channels. In our case, inputs are reduced to a single channel, both to reduce computational complexity and because the task involves calligraphy, which typically uses grayscale intensity and does not benefit from RGB information.

3.4.2. Training Strategy

The models `Calendar Parser` and `Word Parser` are trained on the calendar dataset and the IAM lines dataset, respectively, to develop the capability of producing suitable segmentation maps as output. For both models, the Dice Loss (see Section 3.4) is used during the backpropagation phase to update the weights. The overall learning process is optimized using the Adam optimizer, with an initial learning rate tuned in the range of 0.001 to 0.0001. The batch size for the calendar parser is set to 4, while the best configuration for the word parser is obtained with a batch size of 8 and gradient accumulation set to 16 in order to simulate a larger batch size and improve generalization.

The `Day Digitizer` model is trained on the numeric fonts dataset and fine-tuned on day segments extracted from the calendar via segmentation. The `Month Digitizer`, instead, is trained on a set of month segments obtained through calendar parsing and heavily augmented to improve robustness against issues related to the calendar's month field, as discussed in Section **??**. Both models use the Adam optimizer with an initial learning rate of 0.001 and a batch size of 4.

Table 2
Results of the training achieved by the singular neural components: the best candidate for the calendar segmentation is the one with 6 segmentation classes as output, the month e day digitizers both show great capability in the recognition and so in the classification of the calendar fragments extracted with the segmentation map, word parser is pre-trained over the IAM Lines dataset and becomes good in performing the segmentation of the text line into word pieces and the note digitizer has acquire a good level of digitazion capability even if it should reach a little bit greater performance in terms of CER (Char Error Rate) and WER (Word Error Rate).

Experiment	Valid F1	CER	WER
`Calendar Parser 6`	0.986	-	-
`Calendar Parser 37`	0.906	-	-
`Calendar Parser 49`	0.942	-	-
`Month Digitizer`	0.937	-	-
`Day Digitizer`	0.978	-	-
`Words Parser`	0.986	-	-
`Handwriting Digitizer`	-	0.201	0.272

Table 3
Text classification with BERT LLM: The skills acquired by BERT during its pre-training make it a perfect candidate for text classification. The model can be easily adapted to the new domain with the fine-tuning, and its special context token (i.e. `[CLS]`) allows us to achieve a good level of accuracy in text classification after just few training epochs.

Text Classification	Train F1	Valid F1
`BERT Text Classifier`	0.84	0.77

The `Text Classifier` is trained using Adam as well, with a learning rate of 0.00015 and a batch size of 32, accumulating gradients every 32 steps to simulate an effective batch size of 32×32. This practice, commonly adopted in NLP tasks, is particularly useful when computational resources are limited, as larger effective batch sizes help the model better differentiate between data samples and significantly enhance overall performance.

4. Results

In the current section, the results of the experiment are discussed, highlighting both its strengths and weaknesses. The training outcomes of the Calendar Parser and the other neural modules are reported in Table 2, while the results related to text classification are presented in Table 3.

4.1. Calendar Segmentation

The model performs the segmentation of the calendar in a satisfactory manner. Figure 4 illustrates the six distinct binary masks that are generated when a specific classification segment is requested from the model. Class 0, which corresponds to the None/Contours category, is defined during the training phase because, during manual segmentation of the calendar for ground truth generation, some segments do not overlap, leaving holes in the mask. Class 5 corresponds to the borders of the calendar and is included to ensure that the model focuses on the most relevant features—namely, the one-month, two-day, and three-note segments. Class 4, which corresponds to empty notes, was incorporated into the segmentation map to allow the system to eliminate, during digitization, those areas whose average pixel values are close to 1 (i.e., white). In such cases, it can be inferred that the user has not written any reminders for that day.

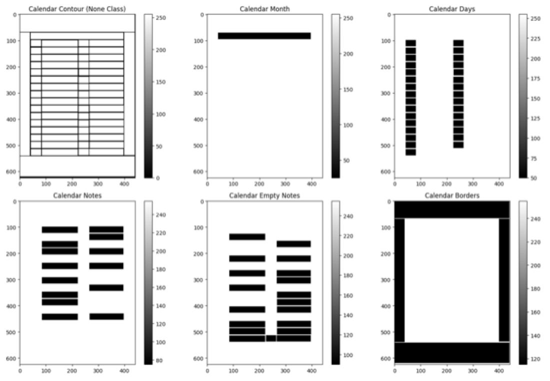


Figure 4: Result of the segmentation process performed on the calendar image. Each binary mask correspond to a particular segmentation value, in the range $[0, 5]$

4.2. Text line segmentation

The Word Parser achieves good accuracy in generating binary segmentation masks to separate sentence images into individual words. The results of this operation are shown in Appendix **??**. As observed there, although the process used to create binary masks—described in Section 3.2—may not always be highly accurate, the model is often able to learn segmentation patterns from the training data that, in some cases, allow it to produce masks even more precise than those generated algorithmically and used as ground truth.

4.3. Calendar fragments digitization

The calendar note digitizer model operates on word segments previously extracted by the `Word Parser`. Each word image is first processed by the encoder, which divides the image into patches, applies positional embeddings, and projects the result into a latent representation. This latent space is then interpreted by the decoder, which maps the encoded features to a sequence of output symbols. In this study, the decoder is constrained to generate output as a sequence of individual characters, which are subsequently reassembled into words.

Although character-level decoding may seem less efficient than full-sentence transcription, it is justified by the limited contextual information available in calendar notes. In general, longer sentences allow the model to capture richer semantic dependencies between words. However, since calendar entries are typically short annotations, the limited context reduces the benefit of sequence-level modeling. In this scenario, it is more appropriate to train the model to learn direct mappings between visual features and their corresponding alphabetic symbols.

This approach yields strong performance at both the character and word levels. Character-level accuracy is measured as:

$$\mathrm{acc}_{char} = 1 - \mathrm{CER}$$

while word-level accuracy is computed as:

$$\mathrm{acc}_{word} = 1 - \mathrm{WER}$$

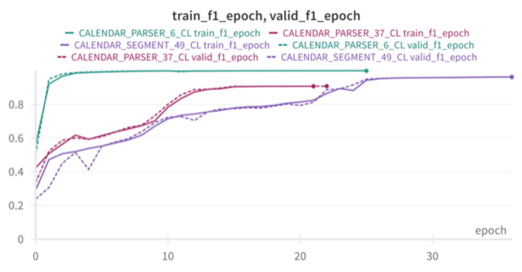
Although the model performs well on the dataset used for pretraining, its performance degrades when applied to our domain—likely due to differences in resolution and handwriting style. We conclude that, to improve accuracy, it is necessary to expand the training dataset with additional handwriting styles and apply further fine-tuning to enhance model robustness.

4.4. Text Content Classification

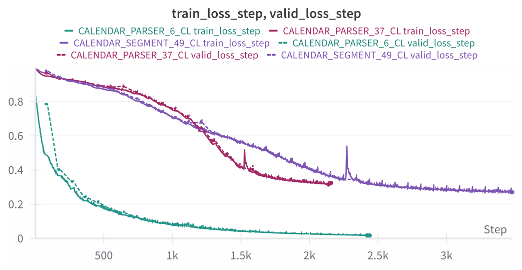
The results of the classification task are reported in Table 3 and appear to be satisfactory. BERT has proven to be an effective tool for achieving the desired outcome, thanks to its ability to leverage semantic knowledge of the English language acquired during pretraining. The classification was performed on a domain consisting of 15 distinct classes. This setting qualifies as a fine-grained classification task, as opposed to a more general coarse-grained classification with broader categories. Under this hypothesis, the trained model can be reused for related classification tasks without requiring retraining. It is possible to cluster fine-grained classes into broader semantic groups and perform classification by mapping each fine-grained label to its corresponding coarse-grained category.

5. Conclusions

Overall, the experiment demonstrates how a character recognition task can be addressed through a sequence of neural models integrated into a single operational



(a) Calendar Segmentation F1 score



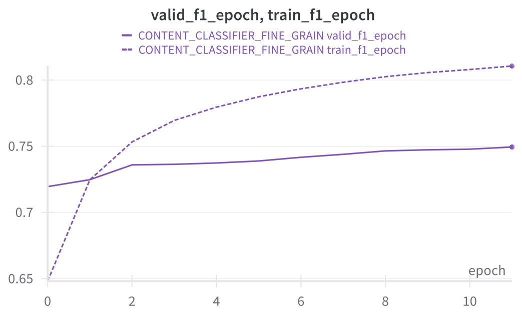
(b) Calendar Segmentation loss

Figure 5: Calendar Segmentation Training Results: from the experiment is it possible to conclude that the model that classification with only six segments class as output has the best performance among the three. This is probably due to the fact the the image structure is too standard.

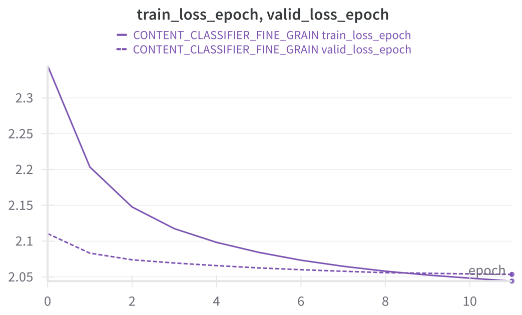
pipeline. The results, particularly in terms of calendar segmentation and subsequent month and day classification, are satisfactory. However, the same cannot be said for the note digitizer, which still exhibits several limitations—both in terms of Word Error Rate and Character Error Rate. A promising direction for improvement would be to enrich the training set with a more heterogeneous handwriting dataset, incorporating a wider variety of calligraphic styles. Indeed, data augmentation alone did not significantly enhance performance when applied to samples that differ substantially from those seen during training and validation. A more diverse dataset could improve the model's ability to extract robust features and, consequently, enhance the overall digitization process.

6. Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT, Grammarly in order to: Grammar and spelling check, Paraphrase and reword. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.



(a) F1 results



(b) Loss Results

Figure 6: Text classification system training results:

References

[1] S. Parsons, C. Parker, C. Chapman, M. Hayashida, W. Seales, Educelab-scrolls: Verifiable recovery of text from herculaneum papyri using x-ray ct, 2023.

[2] E. Stötzner, T. Homburg, J. P. Bullenkamp, H. Mara, R-CNN based PolygonalWedge Detection Learned from Annotated 3D Renderings and Mapped Photographs of Open Data Cuneiform Tablets, in: A. Bucciero, B. Fanini, H. Graf, S. Pescarin, S. Rizvic (Eds.), Eurographics Workshop on Graphics and Cultural Heritage, The Eurographics Association, 2023. doi:10.2312/gch.20231157.

[3] F. Fiani, V. Ponzi, S. Russo, Keeping eyes on the road: Understanding driver attention and its role in safe driving, in: CEUR Workshop Proceedings, volume 3695, 2023, p. 85 – 95.

[4] N. Boutarfaia, S. Russo, A. Tibermacine, I. E. Tibermacine, Deep learning for eeg-based motor imagery classification: Towards enhanced human-machine interaction and assistive robotics, in: CEUR Workshop Proceedings, volume 3695, 2023, p. 68 – 74.

[5] Y. Fujii, [invited] optical character recognition research at google, 2018, pp. 265–266. doi:10.1109/GCCE.2018.8574624.

[6] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, in: Proceedings of the IEEE, volume 86, 1998.

[7] S. Xu, Q. Wu, S. Zhang, S. M. Stanford, Application of neural network in handwriting recognition, 2014. URL: https://api.semanticscholar.org/CorpusID:16468741.

[8] P. Grother, Nist special database 19 handprinted forms and characters database, 1995. URL: https://api.semanticscholar.org/CorpusID:59785963.

[9] G. Lo Sciuto, G. Capizzi, R. Shikler, C. Napoli, Organic solar cells defects classification by using a new feature extraction algorithm and an ebnn with an innovative pruning algorithm, International Journal of Intelligent Systems 36 (2021) 2443 – 2464. doi:10.1002/int.22386.

[10] B. Shi, X. Bai, C. Yao, An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence PP (2015). doi:10.1109/TPAMI.2016.2646371.

[11] M. B. Bora, D. Daimary, K. Amitab, D. Kandar, Handwritten character recognition from images using cnn-ecoc, Procedia Computer Science 167 (2020) 2403–2409. URL: https://www.sciencedirect.com/science/article/pii/S1877050920307596. doi:https://doi.org/10.1016/j.procs.2020.03.293, international Conference on Computational Intelligence and Data Science.

[12] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25, Curran Associates, Inc., 2012.

[13] J. Jebadurai, I. J. Jebadurai, G. J. L. Paulraj, S. V. Vangeepuram, Handwritten text recognition and conversion using convolutional neural network (cnn) based deep learning model, in: 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA), 2021, pp. 1037–1042. doi:10.1109/ICIRCA51532.2021.9544513.

[14] A. Graves, S. Fernández, F. Gomez, J. Schmidhuber, Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks, volume 2006, 2006, pp. 369–376. doi:10.1145/1143844.1143891.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, CoRR (2017). URL: http://arxiv.org/abs/1706.03762.

[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weis-

senborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL: https://arxiv.org/abs/2010.11929. arXiv:2010.11929.

[17] C. Napoli, G. Pappalardo, E. Tramontana, Using modularity metrics to assist move method refactoring of large systems, in: Proceedings - 2013 7th International Conference on Complex, Intelligent, and Software Intensive Systems, CISIS 2013, 2013, p. 529 – 534. doi:10.1109/CISIS.2013.96.

[18] G. Borowik, M. Woźniak, A. Fornaia, R. Giunta, C. Napoli, G. Pappalardo, E. Tramontana, A software architecture assisting workflow executions on cloud resources, International Journal of Electronics and Telecommunications 61 (2015) 17 – 23. doi:10.1515/eletel-2015-0002.

[19] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, CoRR abs/1505.04597 (2015). URL: http://arxiv.org/abs/1505.04597.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009. doi:10.1109/CVPR.2009.5206848.

[21] E. Olson, Apriltag: A robust and flexible visual fiducial system, in: 2011 IEEE International Conference on Robotics and Automation, 2011. doi:10.1109/ICRA.2011.5979561.

[22] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding abs/1810.04805 (2018). URL: http://arxiv.org/abs/1810.04805.

[23] M. A. Fischler, R. C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (1981) 381–395. URL: https://doi.org/10.1145/358669.358692. doi:10.1145/358669.358692.

[24] U.-V. Marti, H. Bunke, The iam-database: An english sentence database for offline handwriting recognition, International Journal on Document Analysis and Recognition 5 (2002) 39–46. doi:10.1007/s100320200071.

[25] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, M. J. Cardoso, Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations, CoRR (2017). URL: http://arxiv.org/abs/1707.03237.

[26] M. Li, T. Lv, J. Chen, L. Cui, Y. Lu, D. Florencio, C. Zhang, Z. Li, F. Wei, Trocr: Transformer-based optical character recognition with pre-trained models, 2022. URL: https://arxiv.org/abs/2109.10282. arXiv:2109.10282.

[27] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.