Interpretable Writer Recognition via Vectors of Locally Aggregated Characters

Tim Raven[1] ✉ [0009−0002−1528−2744], Vincent Christlein[2] [0000−0003−0455−3799], and Gernot A. Fink[1] [0000−0002−7446−7813]

[1] TU Dortmund University, Dortmund, Germany
{tim.raven, gernot.fink}@tu-dortmund.de
[2] FAU Erlangen-Nürnberg, Erlangen, Germany
vincent.christlein@fau.de

Abstract. Writer recognition involves analyzing handwritten documents with respect to the identity of the writer. Although automated methods can achieve strong benchmark results, their lack of interpretability limits practical adoption, particularly in settings where trust and verifiability are critical.
To address this challenge, we propose a novel framework grounded in character-level analysis. At its core is Vectors of Locally Aggregated Characters (VLAC), a feature aggregation method that fuses the aligned outputs of a feature extractor and a feature annotator network. By aggregating local features on a per-character basis, VLAC provides the backbone for computing verifiable character-wise distances, thereby enhancing interpretability and trustworthiness.
We extensively evaluate the proposed framework on two contemporary datasets (CVL and IAM) – achieving new state-of-the-art retrieval results on CVL – as well as a historical dataset (Hist-WI). Our method does not only perform well, but also facilitates interpretable insights into the decision process, paving the way for broader acceptance in practical and forensic applications.

1 Introduction

Writer recognition is the task of analyzing handwritten documents with respect to the identity of the writer. It encompasses several related subtasks:

- **Writer retrieval** aims to retrieve all documents written by the same writer as a given query document from an unlabeled corpus, where the identities of the writers are unknown (neither as personal information nor as abstract writer IDs).
- **Writer identification** compares a query document against a labeled corpus, where writer identities are known. The goal is to assign a writer ID to the query document, either via supervised classification or by treating the task as retrieval and assigning the highest-ranked writer identity.
- **Writer verification** considers a pair of documents and determines whether they were written by the same hand.

These tasks share common methodologies, and our approach is applicable across them without being restricted to a specific subtask.

In recent years, the demand for interpretable AI methods has grown across various domains, including writer recognition. Automated methods achieve high accuracy on benchmark datasets, but their real-world applicability is often hindered by a lack of interpretability. The widespread acceptance depends on the ability to interpret and verify results in a human-understandable manner. This is equally true for forensic experts in legal proceedings as it is for paleographic experts analyzing historical manuscripts.

In this work, we address the interpretability limitations of existing methods by grounding our approach in the analysis of semantic units – specifically, characters. By aggregating local features at the character level, we decompose document distances into character-wise distances, enhancing transparency and interpretability.

To achieve this, we employ two parallel neural networks: a feature extractor and a feature annotator. By aligning their outputs, we obtain character-annotated features, which are subsequently aggregated per character using our novel **Vectors of Locally Aggregated Characters (VLAC)** encoding. The distance between two documents is then computed as the mean cosine distance over these character-wise representations.

Our key contributions are:

1. We introduce VLAC, a novel character-aware feature aggregation method that enhances interpretability while maintaining a strong performance.
2. We achieve state-of-the-art performance on the CVL dataset, containing contemporary handwriting.
3. We demonstrate the robustness of our method by evaluating it on a historical document dataset, achieving promising results.

The remainder of this paper is structured as follows. Section 2 reviews related work on writer recognition. Section 3 details the components of our approach. Section 4 describes the evaluation protocol, including metrics, datasets, and implementation details. Section 5 presents qualitative and quantitative evaluations of our method. Section 6 summarizes our findings and outlines directions for future research.

2 Related Work

Methods for writer retrieval generally follow a common set of stages, namely (1) local feature extraction, (2) feature aggregation and finally (3) distance computation.

Local Feature Extraction. Extracting local features is either done using handcrafted features or deep learning-based features. Examples of handcrafted features used for writer recognition are SURF [13], Zernike moments [3], sparse radial LBP [25] or a combination of SIFT and Pathlets [19]. In recent years, the trend

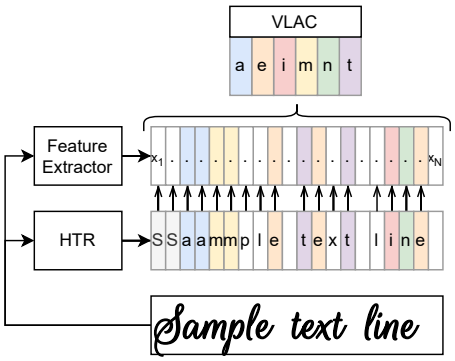


Fig. 1: Illustration of the proposed feature aggregation. Two models are employed in parallel to perform handwritten text recognition (HTR) and to extract local features. By enforcing a one-to-one correspondence between local features and character predictions, the local features can be aggregated by character into the VLAC encoding. Not all characters are encoded, but only those belonging to a refined alphabet $\mathcal{A}^*$, here chosen arbitrarily as {a, e, i, m, n, t}. Note: For visualization purposes only one text line is shown. In practice, local features from all text lines associated with a document are aggregated.

has shifted towards deep learning-based features, learned either by supervised training [10] or self-supervised approaches, such as predicting the cluster index of SIFT descriptors [5], masked image modeling [29, 32], or instance discrimination [26, 32]. While for contemporary data, the use of supervised approaches performs well, on historical datasets self-supervised approaches are dominant.

Feature Aggregation. Feature aggregation composes a document-level descriptor from the extracted local features. The most basic form of aggregation is to directly pool all features (e.g., using sum-pooling). Better performance is usually achieved with a codebook-based feature aggregation. Several such methods have been explored for writer recognition, such as GMM supervectors [4] or Fisher Vectors [9]. However, Vectors of Locally Aggregated Descriptors (VLAD) [14] is the most commonly used, with a range of proposed adaptations: Some methods use several jointly decorrelated VLAD encodings [3, 19] while others include a NetVLAD layer[1] into the feature extraction network[24, 27, 28, 31], and then apply sum-pooling.

Distance Computation. Several distance metrics have been explored in [27]. However, it is most common for methods to use cosine distance. A commonly adopted strategy that has been shown to improve retrieval results is reranking after initial distance computation. Popular examples for reranking methods in the context of writer retrieval include k reciprocal nearest neighbor reranking [15,

31] or graph based reranking using parameter-free message propagation graph networks [28].

Interpretability The authors of [26] investigate the self-attention of different attention heads of a Vision Transformer (ViT) [7] to identify which head attends to which characteristics of the handwriting. The *learnable typewriter* [34] learns to reconstruct text lines from a set of learned sprites. By fine-tuning on individual documents, it can highlight deviations in the typical appearance of a character. In [12] a software tool is presented that can help paleographers by visualizing the similarity of image regions to a given template. Finally, while not interpretable, in [30] a character-level writer retrieval method is presented that only extracts features from certain characters, based on human expert annotations.

3 Method

The core idea of our method is to aggregate local features per-character. This way, the overall distance between two documents can be interpreted via character-wise distances. First, we use a decoder-free HTR model to identify *which* characters appear in each text line and *where* they are located (see Section 3.1). In parallel, we extract local features via a ViT, enforcing a one-to-one correspondence between extracted local features and character predictions (see Section 3.2). We then introduce *Vectors of Locally Aggregated Characters (VLAC)* in Section 3.3, a novel method that groups local features by their predicted character and aggregates them using residuals from global character prototypes. Finally, we compute the distance between two documents by averaging character-specific similarities (see Section 3.4).

3.1 Character Recognition and Localization

To generate a representation that is sensitive to individual characters, it is essential to know both *which* characters occur in a text line and *where* they are located. We achieve this by employing a decoder-free Transformer model $\Theta_{\mathcal{HTR}}$, inspired by the work in [21].

Input Preprocessing and Tokenization. Each input text line is resized to a fixed dimension of 512×64 pixels. A modified ResNet-18 is used for patch embedding, with adjusted strides to partition the text line into 128 tokens. Each token corresponds to a vertical slice of size 4×64 pixels.

Character Prediction. In a single forward pass, the model outputs a sequence of characters $c_{i,j} \in \mathcal{A} \cup \{\epsilon\}$, where $\mathcal{A}$ is the set of viable characters and ϵ is a blank token indicating no character. For a document v consisting of N text lines $l_1, \ldots, l_N$, the model produces:

$$\Theta_{\mathcal{HTR}}(\{l_1, \ldots, l_N\}) = \{c_{i,j} \mid 1 \leq i \leq N,\, 1 \leq j \leq 128\}. \tag{1}$$

The model is trained in a supervised manner using the Connectionist Temporal Classification (CTC) loss [11] (see details in Section 4.3).

Considerations for Model Choice. Many HTR methods use autoregressive encoder-decoder models. In contrast, we use a decoder-free model that predicts the entire output sequence in a single forward pass. We find that this type of model has several advantages for our use-case.

An issue with commonly used autoregressive encoder-decoder models such as TrOCR [20] is their tendency to *hallucinate* when encountering data that is not covered by the training material (for instance, a different language or writing style). Examples for these hallucinations include over-generation of tokens (repeating a character or even a word several times) or spurious outputs where the model produces an entirely different word. The decoder-free Transformer model we use is not autoregressive and produces the entire output sequence in a single forward pass. This mitigates error accumulation during the generation and results in more grounded predictions.

Additionally, each predicted character is directly associated with a specific image region. This eliminates the need for further localization steps, such as analyzing the encoder-decoder attention.

3.2 Feature Extraction

To facilitate the aggregation of local features, we extract a feature vector corresponding to each output token of $\Theta_{\mathcal{HTR}}$.

Local Feature Extraction. We employ a Vision Transformer (ViT) model, denoted by $\Theta_{\mathcal{F}}$, with a patch size of 4×64 pixels, matching the downsampling factor of $\Theta_{\mathcal{HTR}}$. In line with the approach in [32], we discard the class token of the ViT and instead use the final patch tokens as local features. This yields a one-to-one correspondence between the extracted local feature $\mathbf{x}_{i,j}$ and the predicted character $c_{i,j}$ for each token, as shown in Fig. 1. Formally, for a document with N text lines, we define:

$$\Theta_F(l_1, \ldots, l_N) \;=\; \{\, \mathbf{x}_{i,j} \mid 1 \leq i \leq N,\; 1 \leq j \leq 128\}. \tag{2}$$

The ViT is trained in a self-supervised manner using AttMask [17] (see Section 4.3 for implementation details).

3.3 Feature Aggregation

We introduce a novel aggregation method, *Vectors of Locally Aggregated Characters (VLAC)*, which produces a document-level representation by aggregating character-specific features. An illustration is given in Fig. 1.

Character-wise Feature Grouping. Given the one-to-one correspondence between local features and character outputs, we group the features by their corresponding character. For a document v and a character c, let:

$$\mathbf{x}_{v,c} = \{\mathbf{x}_{i,j} \mid c_{i,j} = c\}. \tag{3}$$

Global Character Prototypes. For each character $c \in \mathcal{A}$, we compute a global prototype $\boldsymbol{\mu}_c$ that represents its typical appearance in the training dataset $\mathcal{V}$. Let n_c denote the total number of features annotated as c across $\mathcal{V}$. The global prototype is then computed as:

$$\boldsymbol{\mu}_c \;=\; \frac{1}{n_c} \sum_{v \in \mathcal{V}} \sum_{\mathbf{x} \in \mathbf{x}_{v,c}} \mathbf{x}. \tag{4}$$

Residual Aggregation. To capture document-specific deviations, we aggregate the residuals of the local features relative to the corresponding global prototype. For each character c in document v, the aggregated representation is defined as:

$$\boldsymbol{\Phi}_{v,c} = \ell_2\Big(\sum_{\mathbf{x} \in \mathbf{x}_{v,c}} (\mathbf{x} - \boldsymbol{\mu}_c) \Big), \tag{5}$$

where the ℓ_2-norm ensures that each character representation is independent of how often it occurs in the document.

Character Selection. To improve discriminability, we retain only those characters that contribute effectively to the retrieval task. For each character c, let mAP_c denote the mean Average Precision (mAP) obtained when using $\boldsymbol{\Phi}_{v,c}$ as the document descriptor over the training set. We define a relative threshold τ to form a refined alphabet

$$\mathcal{A}^* = \{c \mid c \in \mathcal{A} \text{ and } \mathrm{mAP}_c \geq \tau \cdot \max_a(\mathrm{mAP}_a)\}. \tag{6}$$

The VLAC descriptor for document v is the collection of the selected character representations:

$$\boldsymbol{\Phi}_v \;=\; \Big\{ \boldsymbol{\Phi}_{v,c} \,\big|\, c \in \mathcal{A}^* \Big\}. \tag{7}$$

3.4 Distance Computation

Given two documents q, v, we first compute the character-wise cosine distances for each character $c \in \mathcal{A}^*$. As the character representations are already ℓ_2-normalized, the cosine distance for character c is

$$d_c(q, v) = 1 - \boldsymbol{\Phi}_{q,c} \cdot \boldsymbol{\Phi}_{v,c}. \tag{8}$$

The distance between the two documents is then given as the average character-wise distance:

$$d(q, v) = \frac{1}{|\mathcal{A}^*|} \sum_{c \in \mathcal{A}^*} d_c(q, v). \tag{9}$$

4 Evaluation Protocol

This section provides an overview of how we evaluate our proposed method. First, in Section 4.1, we cover the different metrics used to assess the performance for different writer recognition tasks. Second, in Section 4.2, we cover the datasets used in our experiments and the preprocessing applied for each dataset. Finally, in Section 4.3 we specify the implementation details used for our experiments unless stated otherwise.

4.1 Metrics

We evaluate our method through a leave-one-out cross-validation, where each test document serves exactly once as the query q. For practical reasons, if a writer contributed only a single document to the dataset, that document is excluded from the query set. We compute the *cosine distance* between the query and each remaining document, ranking them by their distance such that the highest ranked document is the document with the lowest distance. The following metrics are reported:

- **Retrieval (mAP):** We measure retrieval performance by the mean Average Precision (mAP), which evaluates how well the method ranks relevant documents ahead of irrelevant ones.
- **Identification (Top1):** For identification, we use the Top1 accuracy, i.e., the fraction of queries whose highest-ranked document was written by the same writer as the query.
- **Verification (EER):** We measure verification accuracy by the Equal Error Rate (EER), the point where false acceptance and false rejection rates are equal.

4.2 Datasets

We conduct experiments on three datasets: CVL [18], IAM [22], and Hist-WI [8]. Table 1 provides an overview of key statistics for each dataset.

Partitions. For CVL and Hist-WI, we use the official training and test splits. In contrast, IAM does not have widely adopted splits for writer recognition; hence we follow the Aachen split, using its training subset for model training and the remaining documents for testing.

Line Segmentation. Our method operates on text line images. IAM and CVL datasets include pre-segmented text lines, which we use directly. Hist-WI provides only full-page images, so we segment each page into text lines using Transkribus [16].

Binarization. For CVL and IAM, we apply Sauvola thresholding [33] to binarize the segmented text lines. For Hist-WI, we use the provided binarization.

Table 1: Overview of the datasets used for evaluation, showing the number of pages and writers, the distribution of pages per writer (P/W), the contained languages, and the dataset style.

Dataset	Partition	#Pages	#Writers	P/W	Languages	Style
CVL [18]	train	189	27	7	Eng., Ger.	contemporary
	test	1415	283	5	,,	,,
IAM [22]	train	747	283	1 – 59	Eng.	contemporary
	test	792	375	1 – 10	,,	,,
Hist-WI [8]	train	1182	394	3	Lat., Fr., Ger.	historic
	test	3600	720	5	,,	,,

4.3 Implementation Details

Feature Extraction. To train the feature extractor $\Theta_{\mathcal{F}}$ we use AttMask [17], following the proven success for writer retrieval in [32]. We adjust the random resized crop augmentation and only crop in the width direction to achieve the desired scale ($[0.4, 1]$ for global crops, $[0.05, 0.4]$ for local crops), before resizing to 512 or 128 pixel width for global and local crops respectively. $\Theta_{\mathcal{F}}$ is a standard ViT-small, except the patch size being 4×64. We use the training split of Hist-WI and train for 1600 epochs on the roughly 37k extracted text lines with batch size 128, cosine learning rate peaking at $2.5e-3$ after 160 epochs of linear warmup. In line with previous research [31, 32], we use a common model for all datasets.

Handwriting Recognition. To train $\Theta_{\mathcal{HTR}}$, we use the default training recipe proposed in [21] without any adjustments. We employ four Transformer layers with a feature dimensionality of 384. A linear layer maps to the output alphabet of size 80. We train on the IAM training split as outlined in the previous section. We use the same model for all datasets.

Feature Aggregation. We choose a relative threshold for character selection as $\tau = 0.8$ but categorically exclude blank tokens and whitespaces.

Postprocessing. In related work, some form of postprocessing such as PCA and reranking is applied to the final representation. We skip postprocessing by default as it would harm the character-level interpretability. However, as part of our evaluation we investigate the performance of potential postprocessing. In these cases, we use 256 components for the PCA and kRNN reranking with $k = 3$.

5 Experiments

In this section, we present a comprehensive evaluation of our method. First, in Section 5.1, we demonstrate that our CTC-based HTR model is capable of simultaneously recognizing and locating characters. Second, in Section 5.2

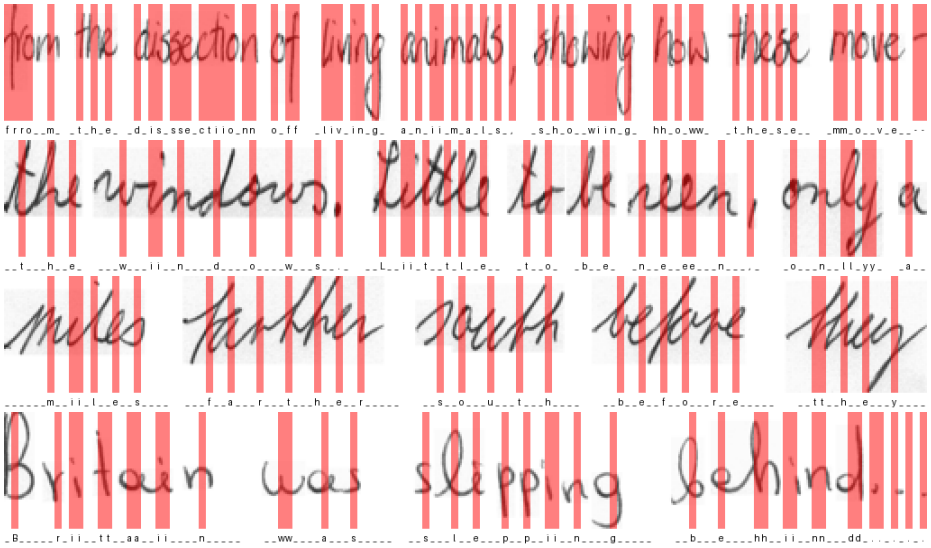


Fig. 2: Visualization of the output alignment of $\Theta_{\mathcal{HTR}}$. Each output token corresponds to a 4×64 pixels slice of the input image. The red bars indicate tokens where a character was predicted (omitting whitespaces). The predicted character is given below the text line. Blank outputs are indicated as '_'. We observe that the character predictions are well aligned with the actual position of the character in the text line.

we compare our novel VLAC encoding to established encodings in different configurations. Third, in Section 5.3, we ablate the components of VLAC to identify the effect on overall performance. Fourth, in Section 5.4, we inspect the character selection step in more detail. Finally, in Section 5.5, we compare the results of our method against prior work.

5.1 Character Recognition and Localization

We use a CTC-style model $\Theta_{\mathcal{HTR}}$ to jointly predict and locate characters. To verify that mapping the token index back to the original image space yields an accurate localization of the character, we plot some sample text lines in Fig. 2 with the corresponding character predictions and localizations. The red vertical bars indicate where a non-blank output was generated. Although we cannot conduct a quantitative analysis because of the lack of character-level annotations, these qualitative results show that the approach works well to implicitly learn a character localization. Our model achieves a character error rate (CER) of 4.9% on IAM and 14.2% on CVL. For Hist-WI we cannot compute the CER as no ground truth annotations are available.

Table 2: Comparison of different encoding strategies. For each encoding, we compare three cases: 1) no postprocessing, 2) PCA whitening fitted on the *training* set, 3) PCA whitening fitted on the *test* set followed by kRNN reranking.

	IAM			CVL/(test)			Hist-WI		
Encoding	Top1↑	mAP↑	EER↓	Top1↑	mAP↑	EER↓	Top1↑	mAP↑	EER↓
Sum	96.2	96.1	1.3	97.7	91.4	3.1	80.0	59.1	15.6
+PCA	96.7	96.8	1.2	98.2	93.7	2.4	87.8	73.5	12.1
+Rerank	92.8	94.4	1.0	99.1	98.6	0.5	86.1	76.7	10.7
VLAD	97.4	97.4	1.2	98.3	93.5	2.6	87.6	71.2	12.0
+PCA	97.4	97.4	0.9	97.6	92.9	2.5	89.5	77.3	11.2
+Rerank	90.2	93.6	0.7	99.4	99.1	0.4	**89.0**	**80.8**	**10.4**
VLAC	98.1	97.8	0.9	99.3	96.3	2.1	87.3	70.5	12.2
+PCA	**98.1**	**98.1**	**0.3**	97.9	95.6	1.5	89.1	76.8	11.7
+Rerank	91.1	94.3	0.4	**99.8**	**99.7**	**0.1**	88.8	80.7	10.5

5.2 VLAC

In this section, we compare VLAC with two well-established encoding methods for writer recognition: sum-pooling and VLAD. For VLAD, we use a codebook of size 50 and apply power normalization ($p = 0.5$). Both VLAD and sum-pooling encode all extracted local features, regardless of their feature annotation. The results are presented in Table 2.

A common approach to improving writer retrieval performance is the use of PCA whitening and reranking. To account for this, we evaluate three configurations:

1. The raw feature vector obtained from the encoding,
2. The feature vector with PCA whitening fitted on the training set,
3. The feature vector with PCA whitening trained on the test set, combined with kRNN reranking. Note that reranking requires access to the entire dataset, so it is consequential to fit the PCA whitening on the test set in this case.

On both contemporary datasets (IAM and CVL), VLAC consistently outperforms VLAD. The improvement is particularly pronounced on CVL, where VLAC improves Top1 accuracy by 1.0%, mAP by 2.8%, and reduces the EER by 0.5%.

Interestingly, reranking significantly degrades performance on IAM, regardless of the encoding method. Similar behavior was observed in preliminary experiments with other reranking techniques, indicating that the issue is not specific to kRNN reranking but rather a more general effect. One possible explanation is the imbalance in the number of pages per writer.

On Hist-WI, VLAC performs slightly worse than VLAD in all three configurations. Given the focus on interpretability, the results are still within a reasonable margin, sacrificing less than 1% mAP, Top1 and EER.

Table 3: Ablation Study investigating the influence of different VLAC design choices: *Proto* indicates whether residuals between local features and character prototypes are aggregated rather than directly aggregating the local features, *Norm* indicates a character-wise ℓ_2 normalization of the aggregated character representations, *Select* indicates whether only a refined set of characters was compared rather than all characters. All results are given in %.

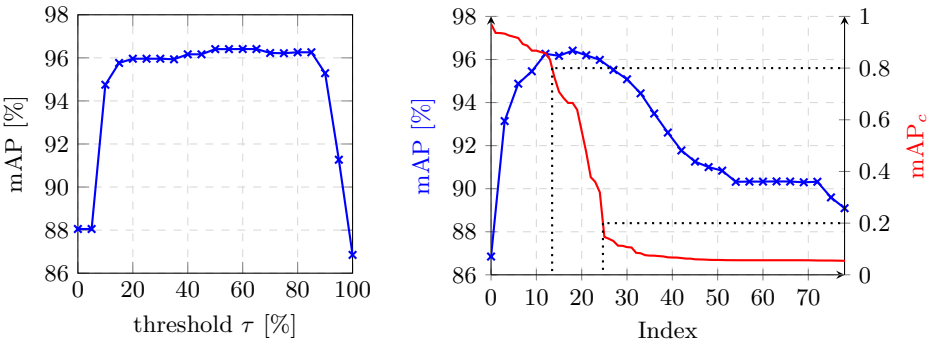
			IAM			CVL/(test)			Hist-WI		
Proto	Norm	Select	Top1↑	mAP↑	EER↓	Top1↑	mAP↑	EER↓	Top1↑	mAP↑	EER↓
			96.0	95.3	1.8	96.2	81.6	7.8	75.1	52.8	18.3
✓			96.4	95.6	3.0	96.8	88.3	6.3	78.6	57.8	15.1
	✓		92.1	90.2	4.7	34.8	25.5	12.1	77.1	56.0	15.6
		✓	97.6	97.1	1.7	94.8	84.4	4.0	85.6	67.2	14.9
✓	✓		**98.1**	**97.9**	1.3	95.5	88.9	4.0	87.0	**70.9**	**12.0**
✓		✓	98.0	97.7	0.9	98.3	94.6	3.2	87.2	70.7	12.6
	✓	✓	**98.1**	97.7	1.7	**99.3**	96.1	**2.0**	85.8	67.4	15.2
✓	✓	✓	**98.1**	97.8	**0.9**	**99.3**	**96.3**	2.1	**87.3**	70.5	12.2

5.3 Ablation Study on VLAC

In this section, we systematically evaluate the influence of various design choices in VLAC. Our analysis focuses on three key components: (1) aggregating residuals between local features and their corresponding character prototypes instead of directly aggregating the raw local features, (2) applying character-wise ℓ_2 normalization to the aggregated residuals (or features), and (3) restricting the aggregation to a subset of characters via a character selection step. When the ℓ_2 normalization is omitted, we concatenate the character representations and compute the cosine distance directly. In total, we assess all eight possible combinations of these components across three different evaluation metrics, with the detailed results presented in Table 3.

For our baseline, we consider the scenario where none of the additional processing steps are employed. Compared to this baseline, the introduction of the character prototype aggregation and the character selection step—when applied in isolation—yields notable performance improvements. In contrast, applying character-wise normalization on its own leads to a significant performance drop on the IAM and CVL datasets, although it proves beneficial on Hist-WI.

Combining any two of the processing steps results in strong synergistic effects, often surpassing the performance gains observed when each component is applied independently. Notably, the combination of character-wise ℓ_2 normalization and character selection consistently enhances performance across datasets, suggesting that these two steps effectively complement each other. The full integration of all three components delivers the most robust and balanced results, even if it does not always deliver the best results.



(a) Threshold Filtering (b) Incremental Inclusion

Fig. 3: Evaluation of character selection strategies. In (a) we include all characters for which mAP$_c$ exceeds a threshold τ on the training set. A very stable performance is observed, indicating the exact choice of τ is not critical. In (b) we include the n highest-ranking characters. The blue line (left axis) indicates the test mAP, whereas the red line (right axis) indicates the ranked single-character mAP (training set). The dotted lines highlight that the choice of cutoff performance has limited effect on the set of retained characters, thus explaining the robustness to varying τ.

5.4 Character Selection

As highlighted by the analysis in the previous section, the character selection step can be essential to achieve robust results. During character selection, we retain only characters within a factor τ of the best single-character performance. To validate this strategy, we evaluate the overall performance at various threshold levels. We focus on the CVL dataset as the influence of the character selection was most noticeable here (see Table 3). We plot the results in Fig. 3a. The influence on performance is minimal. For $\tau \in [0.2, 0.8]$ results are almost equal.

To gain more insight into the results, we incrementally add characters in order of their single-character performance. In Fig. 3b, we plot the performance on the test set when using the n highest-ranked characters, as well as the respective single-character performance of the character at rank n (measured on the train set). As indicated by the dotted lines, the exact choice of cutoff performance has little influence on the set of retained characters. The single-character performance rapidly deteriorates between indices 12 (mAP$_c$ = 85.7%) and 25 (mAP$_c$ = 14.6%). Thus, varying τ has limited impact on the set of retained characters.

Analysis of character utility. As part of the character selection step, we analyze the utility of a character c via the single-character retrieval performance mAP$_c$. In Fig. 4 we plot mAP$_c$ as a function of the normalized character count on the different training datasets. The results show that infrequent characters are not suited for discrimination. Interestingly, sufficiently frequent characters tend

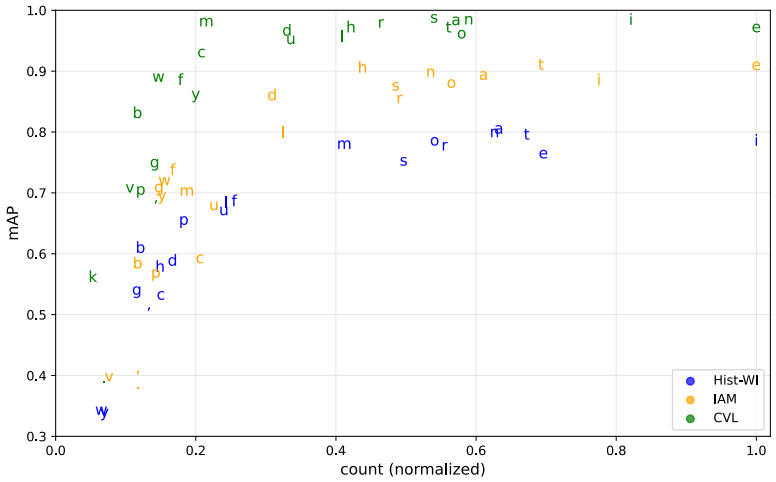


Fig. 4: Analysis of character utility. The horizontal axis shows the normalized count relative to the most frequent character in the dataset. The vertical axis represents the single-character mAP obtained with only this character. For sufficiently frequent characters, no strong correlation between frequency and performance is observable. Note: Due to the CTC-style of our model, there can be multiple features extracted for one occurrence of a character.

to perform comparably well regardless of how often they occur. An especially prominent case is the character *'m'*, which is among the highest-ranked characters in both Hist-WI and CVL despite occurring less frequently than many lower ranked characters. Surprisingly, this does not hold for IAM, highlighting inter-dataset variations.

5.5 Comparison with State of the Art

In this section, we compare our results with established methods. We only consider Top1 accuracy and mAP as previous methods have not computed the EER.

CVL. The results are shown in Table 4. We achieve new state-of-the-art retrieval performance on CVL, outperforming previous methods by more than 1.0% mAP. This is especially impressive as the baseline performance is already very high with 98.6% mAP. However, this also highlights the need for more difficult benchmarks as further improvements are near impossible.

IAM. While other methods have been evaluated on IAM, they did not use a common dataset split [2], and often only consider a subset of the documents (e.g. two documents per writer).

Table 4: Comparison with the state-of-the-art on CVL. *RR* indicates whether reranking was applied.
* in [23] the entire dataset is used for evaluation rather than just the test set.

Method	RR	Top1↑	mAP↑
Mohammad *et al.*[23]		99.8*	-
Christlein *et al.* [6]	✓	99.5	98.4
Rasoulzadeh *et al.* [31]	✓	99.2	98.6
Raven *et al.* [32]	✓	99.4	98.6
Ours	✓	**99.8**	**99.7**

Table 5: Comparison with the state-of-the-art on Hist-WI. *RR* indicates whether reranking was applied.

Method	RR	Top1↑	mAP↑
Christlein *et al.* [5]		88.6	74.8
Peer *et al.* [28]		88.5	73.4
Raven *et al.* [32]		**91.9**	82.6
Ours		87.3	70.5
Peer *et al.* [28]	✓	91.1	80.6
Raven *et al.* [32]	✓	90.9	**83.1**
Ours	✓	88.8	80.7

Hist-WI. As shown in Table 5, our results on Hist-WI fall short of established methods. The results are still promising given the upside of interpretable results.

6 Conclusion

In this paper, we presented VLAC, a novel feature aggregation suitable for interpretable writer recognition. By aggregating local features on a character level, the overall distance between two documents can be expressed in terms of individual characters. This allows experts to potentially interpret and verify the results.

In our experiments, we showed that VLAC is a very powerful encoding for writer recognition, achieving robust results on several datasets. Especially on contemporary datasets, where the HTR results are very reliable, VLAC outperforms other encodings. We showed that when including a reranking step, our method achieves state-of-the-art retrieval performance while also achieving near perfect Top1 accuracy (99.8%) and EER (0.1%). On the IAM dataset, we also achieve very strong results. On the historical Hist-WI dataset, our results fall short of other methods. On the one hand, this may be due to insufficient HTR performance. Training on IAM and evaluating on Hist-WI is a significant domain shift, both in terms of script type as well as the featured languages. With a more suitable training set for the HTR task, the performance of VLAC would likely improve. On the other hand, even when using VLAD our results fall short of those reported in [32] despite a very similar feature extraction. This may be related to us extracting fewer descriptors, or to us working on line level, or to the different tokenization using vertical stripes rather than patches. More research is needed in this direction.

Our work also opens up many other avenues for further research. Firstly, other mechanisms for feature annotation could be explored, for instance, based on the cross-attention of an encoder-decoder model. The alignment is more difficult in this setting, but it would allow for greater flexibility with respect

to the tokenization process. Secondly, it may be possible to unify the feature extraction and HTR models into a single model, potentially trained end-to-end for transcription and writer recognition. Finally, our results highlight the need for more challenging benchmarks for writer recognition tasks on contemporary data.

On a different note, future research is also needed in terms of practical application of VLAC. Our results have primarily been focused on an empirical level. More work is needed to verify that the computed character-level distances align with the judgment of human experts.

References

1. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: Cnn architecture for weakly supervised place recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5297–5307 (2016)
2. Christlein, V.: Handwriting analysis with focus on writer identification and writer retrieval. Friedrich-Alexander-Universitaet Erlangen-Nuernberg (Germany) (2019)
3. Christlein, V., Bernecker, D., Angelopoulou, E.: Writer identification using VLAD encoded contour-Zernike moments. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR). pp. 906–910 (Aug 2015). https://doi.org/10.1109/ICDAR.2015.7333893
4. Christlein, V., Bernecker, D., Hönig, F., Angelopoulou, E.: Writer Identification and Verification using GMM Supervectors. In: IEEE Winter Conference on Applications of Computer Vision. pp. 998–1005. IEEE (2014)
5. Christlein, V., Gropp, M., Fiel, S., Maier, A.: Unsupervised Feature Learning for Writer Identification and Writer Retrieval. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 1, pp. 991–997. IEEE (2017)
6. Christlein, V., Maier, A.: Encoding CNN Activations for Writer Recognition. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS). pp. 169–174. IEEE (2018)
7. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In: International Conference on Learning Representations (2021)
8. Fiel, S., Kleber, F., Diem, M., Christlein, V., Louloudis, G., Nikos, S., Gatos, B.: ICDAR2017 Competition on Historical Document Writer Identification (Historical-WI). In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 1, pp. 1377–1382. IEEE (2017)
9. Fiel, S., Sablatnig, R.: Writer Identification and Writer Retrieval using the Fisher Vector on Visual Vocabularies. In: 2013 12th International Conference on Document Analysis and Recognition. pp. 545–549. IEEE (2013)
10. Fiel, S., Sablatnig, R.: Writer Identification and Retrieval using a Convolutional Neural Network. In: Computer Analysis of Images and Patterns: 16th International Conference, CAIP 2015, Valletta, Malta, September 2-4, 2015, Proceedings, Part II 16. pp. 26–37. Springer (2015)
11. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks.

In: Proceedings of the 23rd international conference on Machine learning. pp. 369–376 (2006)
12. Grieggs, S., Henderson, C.E.M., Sobecki, S., Gillespie, A., Scheirer, W.: The paleographer's eye ex machina: Using computer vision to assist humanists in scribal hand identification. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 7177–7186 (January 2024)
13. Jain, R., Doermann, D.: Combining local features for offline writer identification. In: 2014 14th International Conference on Frontiers in Handwriting Recognition. pp. 583–588. IEEE (2014)
14. Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating Local Image Descriptors into Compact Codes. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(9), 1704–1716 (2011)
15. Jordan, S., Seuret, M., Král, P., Lenc, L., Martínek, J., Wiermann, B., Schwinger, T., Maier, A., Christlein, V.: Re-ranking for writer identification and writer retrieval. In: Document Analysis Systems: 14th IAPR International Workshop, DAS 2020, Wuhan, China, July 26–29, 2020, Proceedings 14. pp. 572–586. Springer (2020)
16. Kahle, P., Colutto, S., Hackl, G., Mühlberger, G.: Transkribus - a service platform for transcription, recognition and retrieval of historical documents. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). vol. 04, pp. 19–24 (2017). https://doi.org/10.1109/ICDAR.2017.307
17. Kakogeorgiou, I., Gidaris, S., Psomas, B., Avrithis, Y., Bursuc, A., Karantzalos, K., Komodakis, N.: What to Hide from Your Students: Attention-Guided Masked Image Modeling. In: European Conference on Computer Vision. pp. 300–318. Springer (2022)
18. Kleber, F., Fiel, S., Diem, M., Sablatnig, R.: CVL-Database: An Off-Line Database for Writer Retrieval, Writer Identification and Word Spotting. In: 2013 12th International Conference on Document Analysis and Recognition. pp. 560–564. IEEE (2013)
19. Lai, S., Zhu, Y., Jin, L.: Encoding Pathlet and SIFT Features With Bagged VLAD for Historical Writer Identification. IEEE Transactions on Information Forensics and Security **15**, 3553–3566 (2020)
20. Li, M., Lv, T., Cui, L., Lu, Y., Florencio, D., Zhang, C., Li, Z., Wei, F.: TrOCR: Transformer-based optical character recognition with pre-trained models. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 12814–12822 (2023). https://doi.org/10.1609/aaai.v37i11.26538
21. Li, Y., Chen, D., Tang, T., Shen, X.: Htr-vt: Handwritten text recognition with vision transformer. Pattern Recognition **158**, 110967 (2025)
22. Marti, U.V., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. International Journal on Document Analysis and Recognition **5**, 39–46 (2002)
23. Mohammed, H., Märgner, V., Konidaris, T., Stiehl, H.S.: Normalised local naïve bayes nearest-neighbour classifier for offline writer identification. In: 2017 14th IAPR international conference on document analysis and recognition (ICDAR). vol. 1, pp. 1013–1018. IEEE (2017)
24. Ngo, T.T., Nguyen, H.T., Nakagawa, M.: A-vlad: An end-to-end attention-based neural network for writer identification in historical documents. In: International Conference on Document Analysis and Recognition. pp. 396–409. Springer (2021)
25. Nicolaou, A., Bagdanov, A.D., Liwicki, M., Karatzas, D.: Sparse radial sampling lbp for writer identification. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR). pp. 716–720. IEEE (2015)

26. Peer, M., Kleber, F., Sablatnig, R.: Self-supervised Vision Transformers with Data Augmentation Strategies Using Morphological Operations for Writer Retrieval. In: International Conference on Frontiers in Handwriting Recognition. pp. 122–136. Springer (2022)
27. Peer, M., Kleber, F., Sablatnig, R.: Writer retrieval using compact convolutional transformers and netmvlad. In: 2022 26th International Conference on Pattern Recognition (ICPR). pp. 1571–1578. IEEE (2022)
28. Peer, M., Kleber, F., Sablatnig, R.: Towards Writer Retrieval for Historical Datasets. In: International Conference on Document Analysis and Recognition. pp. 411–427. Springer (2023)
29. Peer, M., Kleber, F., Sablatnig, R.: Saghog: Self-supervised autoencoder for generating hog features for writer retrieval. In: International Conference on Document Analysis and Recognition. pp. 121–138. Springer (2024)
30. Peer, M., Sablatnig, R., Serbaeva, O., Marthot-Santaniello, I.: Kairacters: Character-level-based writer retrieval for greek papyri. In: International Conference on Pattern Recognition. pp. 73–88. Springer (2025)
31. Rasoulzadeh, S., BabaAli, B.: Writer identification and writer retrieval based on NetVLAD with Re-ranking. IET Biometrics **11**(1), 10–22 (2022)
32. Raven, T., Matei, A., Fink, G.A.: Self-supervised vision transformers for writer retrieval. In: International Conference on Document Analysis and Recognition. pp. 380–396. Springer (2024)
33. Sauvola, J., Seppanen, T., Haapakoski, S., Pietikainen, M.: Adaptive Document Binarization. In: Proceedings of the Fourth International Conference on Document Analysis and Recognition. vol. 1, pp. 147–152. IEEE (1997)
34. Siglidis, I., Gonthier, N., Gaubil, J., Monnier, T., Aubry, M.: The learnable typewriter: a generative approach to text analysis. In: International Conference on Document Analysis and Recognition. pp. 297–314. Springer (2024)