1. Style classification

1.1. Materials

Two datasets were used for the purpose of style classification. First, a dataset containing extracted graphemes of different allographs extracted from documents dated to three different periods Dead Sea Scross writing. This dataset contained a total of 3099 graphemes, each labeled with one of the 27 letters and one period (349 Archaic, 1372 Hasmonean and 1378 Herodian). Some examples are given in fig. 1.1.

Figure 1.1.: Examples of the labeled graphemes. From left to right: the letters Lamed, Samekh and Shin, for each letter three versions are given in the Archaic, Hasmonean and Herodian styles (again left to right)

In addition to these graphemes, a number of images of complete fragments binarized with BiNet were used. These fragments were partially annotated by style, corresponding to the three classes mentioned above. The fragments were labeled as follows: Archaic (2), Hasmonean (3), Herodian (4), unlabeled (20).

1.2. Methods

The style classification consists of two stages: first a codebook is learned from a set of annotated allograph images (see fig. 1.2) and then this codebook is used to determine the pairwise similarity of documents are in terms of style (fig. 1.4). If a collection of reference documents annotated by style is available, the most similar reference images for an unseen fragment can be retrieved and used to predict its style.

To construct the codebook, a fraglet representation of graphemes in the images was used. Fraglets are parts of a connected component (2004), and are represented in this case by the contour surrounding part of the connected component. The contour is a closed curve in pixel coordinates sampled in a fixed number of evenly spaced points, leading to a feature vector of fixed dimensionality. A dimenstion reduction method is used to project the contour features to a lower-dimensional space in a way that preserves the data topology, with a metric that rewards keeping different styles of the same allograph apart. The reduced-dimension fraglets are clustered to construct a codebook of characteristic shapes. Fragments that are to be classified are encoded by assigning each

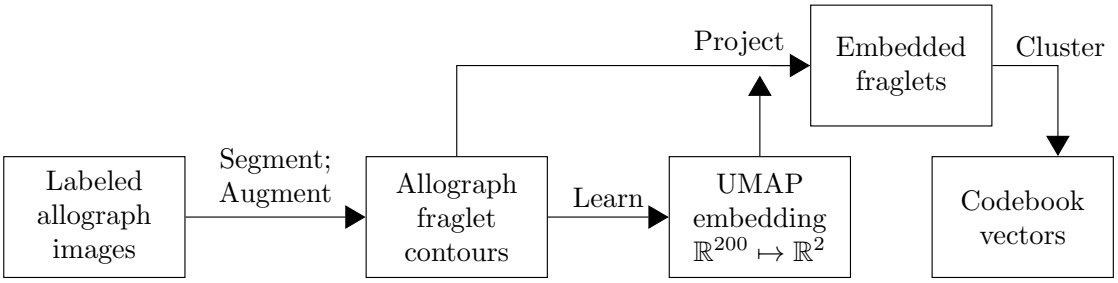


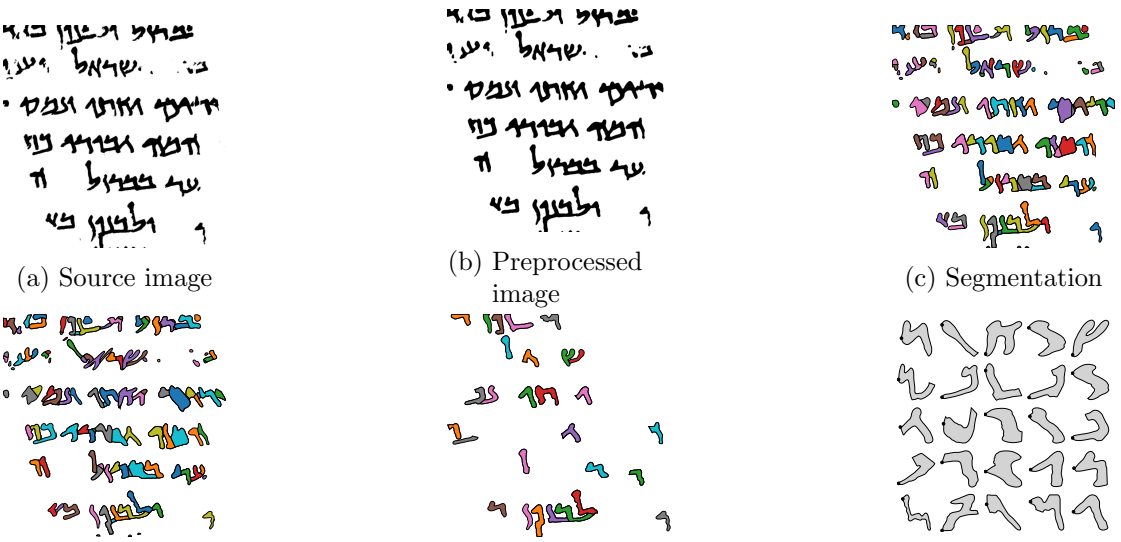
Figure 1.2.: Schematic overview of the codebook construction process

fragment to the nearest codebook point and interpreting the distribution as a probability density function.

To be able to deal uniformly with dat with various levels of annotation (source image, style label, allograph label) a processing pipeline was set up to process batches of images and store annotated sets of fraglets in the NetCDF format using the `xarray` Python package (Hoyer and Hamman 2017).

1.2.1. Contour extraction and augmentation

The first part of the pipeline for processing either fragments (consisting of several lines of text) or allographs (containing a cutout of a single allograph) is the extraction of fraglets from the image. The steps of this process are shown in figure 1.3.



(a) Source image (b) Preprocessed image (c) Segmentation

(d) Augmentation (e) Filtering (f) Normalization

Figure 1.3.: The different processing steps going from fragment to contour

The input images are binarized with BiNet (Dhali, Wit, and L. Schomaker 2019), an image-level Gaussian filter with a standard deviation of 2 pixels is applied, and a

morphological closing with a disk of radius 1 as a structuring element is performed. The images are then thresholded with the otsu-method (Otsu 1979). After this, connected components of the image are found, holes in connected components are filled, and for each connected component a clockwise contour is calculated using an algorithm that traces the contour boundary at pixel level. When data augmentation is applied, augmented contours are rotated with a rotation angle drawn from $\mathcal{N}(0, 5\,\mathrm{deg})$ and sheared horizontally with a shear factor drawn from $\mathcal{N}(0, 0.3)$. in order to generalize over the writing slant and line orientation. The are then morphed with a displacement where local displacements are drawn from a normal distribution with a standard deviation of 20 pixels and spatially correlated over a distance of 40 pixels.

A novel method was used to segment connected components into fraglets. For each contour extracted from the binarized fragment image, the leftmost and rightmost point is found. The contour is split into a top half and a bottom half at these points, and the line from left to right that is equidistant from both halves is found by calculating the euclidean distance transform of both halves and tracing a curve through the points where both distances are equal. Then split points are found along this midline using a heuristic that is calculated in each midline point. This heuristic is the sum of the hinge of the top contour and the derivative of bottom-contour index with respect to midline contour index. The contours are split at the local maxima of this heuristic value.

Extracted contours are normalized by putting the start point in the leftmost point, substracting the centroid and dividing the x- and y-coordinates by their standard deviations. They are then interpolated to 100 evenly spaced points. Fraglets with a low area or a low periphery compared to the area are removed in order to keep only fraglets with a clear shape.

1.2.2. UMAP embedding

In order to quantify the difference between a pair of fragments, a dimension reduction is required. In order to be able to preserve the information required for style classification, it is imporant that the reduced-dimension representation of the fraglets still contains enough information to distinguish between different styles of the same letter. To achieve this, a nonlinear dimension reduction is performed using the *Universal Manifold Approximation and Projection* (UMAP) method (McInnes et al. 2018). This method can learn a projection from the fraglets to a lower-dimensional space in a semisupervised way, combining information from the shape of the fraglets with information from the style labels. As a shape metric, the euclidean distance between a vector containing all contour points is used. The supervision metric penalizes points corresponding to different styles when they are close together, and does not take into account the temporal ordering of the styles.

1.2.3. Codebook creation

A codebook is created by performing K-means clustering on the embedded graphemes. The centroid of each cluster is used as a codebook vector. Fraglets from a fragment that

should be classified are assigned to the nearest codebook vector.

1.2.4. Fragment classification

For fragment classification, the fraglets in a fragment are quantized using the codebook and the normalized distribution over the codebook vectors is calculated (see fig.1.4). This distribution is interpreted as a probability denisty function over the codebook (codebook pdf). The χ^2-distance between codebook pdfs is used to determine the style similarity of fragments. Style clasification is done by finding the most similar fragment among a set of example documents that are labeled by style. The method might not generalize well to fragments that differ significantly from these labeled examples.

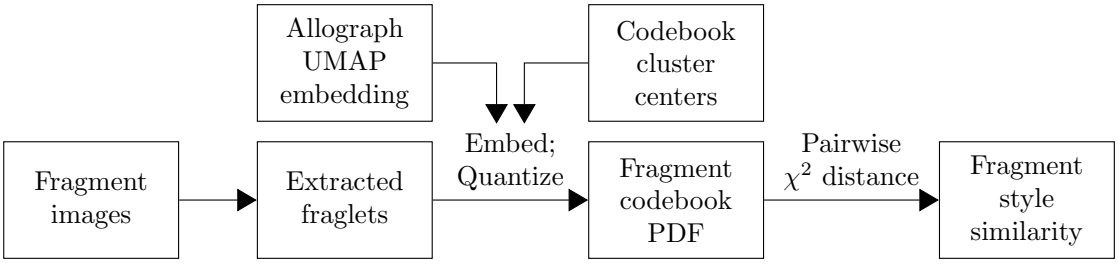


Figure 1.4.: Determining style similarity of fragments

1.3. Results

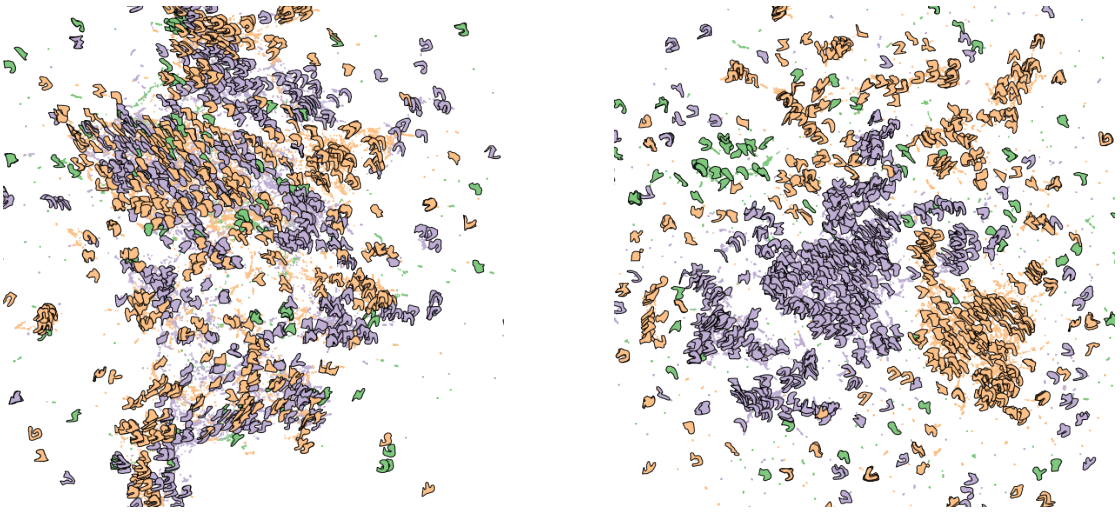
A grid search over parameters was performed. The grid search was done on a laptop computer with a total run-time of about one hour. In the table in the appendix, results are given for several values of the parameters w_{style}, which controls the relative importance of separating the style classes compared to that of preserving the data topology, N_{dim} (the number of dimensions in the projection space) and $N_{codebook}$ (the size of the codebook). In figure 1.5, two embeddings are shown with different values of w_{style}.

For each row, three embeddings were learned for the codebook and used to find the codebook PDF for each of the fragments. To quantify the separation between the styles in the codebook, for the fraglets quantized to each codebook vector the homogeneity of the style classes is calculated using the method of Rosenberg and Hirschberg 2007. This score is equal to one if every codebook vector contains only fraglets annotated with the same style.

The unlabeled fragments were split in a top and bottom half containing roughly the same amount of text in order to obtain two fragments that do not share graphemes but should share the same style. Mean inter-fragment and intra-fragment distances and their ratio are reported in the grid search table. For the labeled fragments, the nearest neighbour was found for each fragment, and its syle label was used as a predicted label. The style accuracy is calculated as the fraction of these fragments where the predicted style label matches the ground-truth label. This was repeated once without agmentation

and once with data augmentation increasing the number of fraglets for each fragment by a factor of 5.

In table 1.1 the confusion matrix is given for a classifier trained with $w_{\text{style}} = 0.15$, $N_{\text{neigbours}} = 20$, $N_{\text{codebook}} = 150$, and in figure 1.6 a distance matrix is shown for the same classifier.



(a) $w_{\text{style}} = 0.15$ (b) $w_{\text{style}} = 0.45$

Figure 1.5.: Examples of embeddings with a low and a high style separation weight ($N_{\text{neighbours}} = 20$, $N_{\text{dim}} = 2$). Scattered points indicate codebook vectors, overlayed contours show the shape of 1% of the fraglets, placed at the point where they are projected by the UMAP embedding. The colors indicate the style of that codebook allograph (green is Archaic, purple is Hasmonean, orange is Herodian)

	Predicted label		
True label	Archaic	Hasmonean	Herodian
Archaic	0.7	1.1	0.2
Hasmonean	0.0	2.4	0.6
Herodian	0.1	1.8	2.1

Table 1.1.: Confusion matrix for style classification, averaged over 10 codebook clusterings with 1 embedding

Bibliography

Dhali, Maruf A., Jan Willem de Wit, and L. Schomaker (2019). "BiNet: Degraded-Manuscript Binarization in Diverse Document Textures and Layouts using Deep Encoder-Decoder Networks". In: *ArXiv* abs/1911.07930.

Hoyer, S. and J. Hamman (2017). "xarray: N-D labeled arrays and datasets in Python". In: *Journal of Open Research Software* 5.1. DOI: 10.5334/jors.148. URL: http://doi.org/10.5334/jors.148.

McInnes, Leland et al. (2018). "UMAP: Uniform Manifold Approximation and Projection". In: *The Journal of Open Source Software* 3.29, p. 861.

Otsu, Nobuyuki (1979). "A Threshold Selection Method from Gray-Level Histograms". In: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1, pp. 62–66. DOI: 10.1109/TSMC.1979.4310076.

Rosenberg, Andrew and Julia Hirschberg (June 2007). "V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure". In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic: Association for Computational Linguistics, pp. 410–420. URL: https://aclanthology.org/D07-1043.

Schomaker, Lambert and Marius Bulacu (June 2004). "Automatic Writer Identification Using Connected-Component Contours and Edge-Based Features of Uppercase Western Script". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 26.6, pp. 787–798. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2004.18. URL: https://doi.org/10.1109/TPAMI.2004.18.