Part IV

STYLE CLASSIFICATION AND DATING

How to date the undated?
With the success of writer identification in Chapter 4, we continue our work in this chapter, where major style classification using self-organizing maps on the character level of DSS is explored.

While writer identification focuses on connecting a manuscript to a specific individual, dating of handwriting looks at a broader temporal context to provide a range of periods during which the writing could have been produced. Even though the end goal differs, both domains can benefit from each other and share some of the feature extraction and pattern recognition techniques.

However, before proposing a date prediction model in the next Chapter 6, this chapter examines and presents the results of classifying manuscripts as being a member of large (historical) style categories. Again, new challenges of within- and between-class variance come into play. The work of this chapter finally lays the ground for our final date prediction model in the latter chapter.

PROVENANCE
This chapter is an adaptation of an article that has been previously published as:

AUTHOR CONTRIBUTIONS
Dhali conceived the complete setup, performed and moderated the experiments, performed the analysis with other co-authors, wrote the draft, and wrote the final manuscript.
5.1 Introduction

In the study of historical manuscripts, scholars often explore four major questions: what was written, by whom, when, and where \[274\]. These questions help in understanding the historical context. In this chapter, we focus on the ‘when’ question, i.e., the dating of the manuscripts. Dating a historical handwritten manuscript requires the inference of expert palaeographic scholars. The scholars rely on their knowledge and experience to estimate the date for a particular manuscript. Several aspects are taken into consideration, including the writing style, the contents, and even the writing materials. This process requires a lot of time and human effort. Furthermore, due to the subjective nature of these approaches, contrasting opinions for an estimated date are always on the table. An automatic system based on modern pattern recognition techniques would be the best tool for the palaeographers helping them to assess hypotheses along with providing new ones. In this study, an important collection of historical handwritten manuscripts, the Dead Sea Scrolls (DSS), is studied to find the chronological style development of the handwriting.

The DSS collection contains damaged scrolls and fragments discovered in the mid-20th century in the Judaean desert near the Dead Sea. These scrolls contain, among others, the oldest known biblical manuscripts, and hold tremendous religious and historical value. The DSS is written in characters of the Hebrew alphabet, which is derived from the older Aramaic script \[316\]. Most of the scrolls were written over an estimated time period of almost four centuries (ca. 250 BCE to ca. 135 CE), by multiple different writers \[214, 287\]. The time span of the scrolls is traditionally, following the work of Frank Moore Cross, subdivided into three main periods. In sequence, they are Archaic, Hasmonean, and Herodian \[55\]. The number of internally dated manuscripts is very low in the DSS collection. The dates of most of the manuscripts have not been recorded at the time of their production. Efforts have
been made by scholars to determine the dates of the scrolls using human assessment of writing style and pragmatic considerations on provenance and material. Although the radiocarbon dating method was already developed almost at the same time that the scrolls were discovered, only a few tests have been carried out since then [25, 241]. In addition, within the framework of the European Research Council (ERC) project "The Hands that Wrote the Bible", new radiocarbon samples for the DSS are being processed and prepared for publication [75]. However, radiocarbon dating can only be performed on a very limited number of physical samples due to the method’s destructive nature. Therefore, it is important to develop a pattern recognition-based framework for dating which will be able to accommodate both human knowledge and radiocarbon dates. Initial research on writer identification has been done on the DSS collection using several feature extraction techniques to analyze differences in handwriting style among manuscripts to determine the writer [65]. This chapter is a continuation of the ongoing research work on the DSS and will focus on the dating of the scrolls using pattern recognition techniques.

The field of digital palaeography is concerned with achieving an automatic dating process of the manuscripts. This process requires digital images. In recent times, there have been many efforts to produce digitized sets of historical manuscripts, as to enable scientific research on them. One of the early digitized sources of the DSS is from Brill Publishers [165], containing more than two thousand images. Another example is the Medieval Palaeographic Scale (MPS) dataset, containing medieval charters from the period 1300-1550 CE. This dataset was used by [109] to test their historical manuscript dating framework based on handwritten pattern analysis. Another dataset of the medieval charters called the Svenskt Diplomatariums huvudkartotek (SDHK) was digitized and used in the work of [304] to test their Convolutional Neural Network (CNN) for historical manuscript dating. The manuscripts from these latter two sets originate from Europe and are written in Roman script. The real dates for the manuscripts in these datasets were recorded, making them excellent datasets to develop and test new dating models on. On the contrary, the amount of labeled manuscripts in the DSS collection is extremely low. Dating these manuscripts even poses a further challenge due to their damaged condition.

Several different approaches have been developed for digital historical manuscript dating to aid scholars with improved performance. Two major approaches that have been successful in achieving this are deep learning-based and dedicated feature methods based. In general, for a deep learning-based approach, large amounts of labeled data are required. With historical manuscript dating, these large amounts of labeled data are often not available. To solve this problem, the pre-trained Google ImageNet network was used as a starting point for a model [304]. The SDHK dataset was used to further train and test their model. Even though the amount of training data was lower in SDHK than what is generally required to train a CNN, it is still a lot higher than the DSS-labeled data. A feature-based pattern analysis approach for
handwritten manuscripts handles low amounts of data better. The rationale behind style-based dating is that over time, the general handwriting style of people changes, and slight variations in the way characters are written start to occur. By modeling these changes, a date can be predicted. To extract the handwriting patterns from a document, a feature extraction method is needed that can map the raw pixels of an image to a descriptive feature vector. In the work of [109], a grapheme-based feature extraction method was used in combination with a temporal pattern codebook to achieve dating results on the MPS dataset. Multiple textural methods were proposed that achieved varying results dating the MPS dataset as well [115].

The aim of this study is to understand if a handwriting pattern-based dating approach on the DSS can achieve results similar to the estimated dates that have been proposed by scholars. On top of that, an accurate estimation by the system provides a tool for confirming or revising the rough periodization of the mentioned timeline. In order to build the system, this chapter will explore the dedicated feature extraction methods on a selection of the DSS collection and provide an evaluation of their performances. Though the processing of the entire collection of the DSS poses a higher challenge than most of the datasets containing historical handwriting manuscripts, this work would be the framework for further research on the style-based chronological development of the DSS.

5.2 Methodology

5.2.1 Data

Figure 5.1: Full spectrum color images of two fragments with color calibrators, scale bars, plate labels, and the adhesion tapes (left: plate 134, fragment 20; right: plate 235, fragment 1).

In this study, the most recent digitized images of the DSS from the IAA are used. The multi-spectral images of the scroll fragments on the recto and verso in 28 different exposures have been captured. The resolution of the images is 1,215
pixels per inch at a 1:1 ratio [259]. The images were captured by putting the scroll fragments against dark backgrounds. In addition to the fragment, the images contain color calibration strokes, a plate number label, a scale bar, and sometimes adhesion tape. The fragments themselves have different background materials, and they vary wildly in their quality. Most of them are heavily damaged, containing missing parts and holes. A couple of examples of fragments can be seen in Figure 5.1. For some of the fragments, the degradation leads to low contrast between the ink and the background material, making them hard to read. Making use of the different spectra by using a band where the contrast between the ink and the background is stronger, can provide a way to distinguish between the ink and the background more easily. The spectra can also provide underlying information on the background material or other textual properties, as these attributes can be more pronounced in the specific spectrum.

5.2.2 Ground truths

In the DSS collection from the IAA, every fragment image has an associated plate number label, referencing the plate they belong to. A plate can contain multiple fragments. These plates have a simple numbering system starting from 1 and incrementing, with some exceptions. The fragments of the same plate do not necessarily form the full manuscript. Usually, Q-numbers are used to indicate a particular manuscript, and they have the format as 'prefix-Q-suffix'. The prefix indicates the cave number, and the suffix indicates the manuscript number. An example of a Q-number is 4Q288, indicating cave 4 and manuscript 288. For testing purposes, in the ERC project, we have categorized the manuscripts on the IAA plates according to the traditional nomenclature for dating periods. These periods are, in sequence: Archaic, early-Hasmonean, Hasmonean, late-Hasmonean, early-Herodian, Herodian, late-Herodian, and post-Herodian. Typically, Archaic ranges from 300-175 BCE, Hasmonean from 175-40 BCE, with early 175-100 BCE and late 100-40 BCE, and Herodian from 40 BCE - 70 CE, with early 40 BCE - 10 CE and late 10-70 CE, and post-Herodian for 70-135 CE. Post-Herodian is not considered in this study due to the very low number of labeled manuscripts. Manuscripts labeled only as Hasmonean or Herodian are less specific in their estimation, as these encompass the whole period instead of the early or late part. One important note here is that these ranges are not exact; rather, an estimation. A discussion on the exactness of these periods is beyond the scope of this work. These ranges will act as data points only, and will not have any impact on the framework of the model. Changing these date ranges will always be possible following scholarly consensus.
5.2.3 Preprocessing

In order to perform feature extraction, a binarized image is necessary where only the relevant ink parts are visible. Irrelevant materials on the image, e.g., the color calibrators and plate labels, need to be removed. In the binarization step, each pixel is threshold to either be a background (white) pixel or a foreground (black) pixel. The goal is to have only the relevant ink parts marked as foreground pixels. A method commonly used for binarization is Otsu’s binarization [203]. It is often chosen due to it being efficient and non-parametric. This method generally works quite well, but in the DSS, many fragments have low contrast between their background materials and the ink, causing the binarization to fail to produce the desired results. Additionally, some fragments are leather-based, with skin texture, whereas others are written on papyrus with a repetitive fiber pattern. Ink traces may have lost tiny flakes due to desiccation or were not filled properly due to imperfect absorption by the surface material at the time of writing. Because of these considerations, a method is required that is more suited for these images. A binarization method using neural networks was developed for the DSS. This method, BiNet, was presented in Chapter 3. In Figure 5.2, the results of this binarization technique can be seen for a fragment image with low contrast between the ink and the background. The binarized images will be used as the input for the feature extraction.

Figure 5.2: Full spectrum color image of a fragment on the left and the corresponding binarized image on the right (plate 238, fragment 1).

5.2.4 Feature extraction techniques

The methodology is based on the idea that the handwriting style of the general population evolves over time. By capturing this change over time, the general style of each time period can be determined. This can then be used to make inferences on a manuscript’s date, comparing its handwriting style to the general styles of the time period. To extract the handwriting styles, a feature extraction method is
needed that captures the handwriting style in a feature vector. Two common groups of feature extraction methods that will be explored in this study are textural-based and grapheme-based. One grapheme-based method and seven textural methods are explored.

5.2.4.1 Textural methods

Textural-based methods consider the texture of the handwriting patterns on the binarized image of a manuscript. These methods capture statistical information on attributes of handwriting, like curvature and slant of the contours. As these methods look at the image as a whole, they do not require a segmentation technique. The statistical information is captured in a feature vector that represents the handwriting style used in the manuscript and can be used for further analysis. There are several types of textural methods. These can roughly be categorized into contour-based methods and filter-based methods. In this study, only contour-based methods are explored.

In the work of [34], the Hinge kernel and corresponding Hinge feature were proposed. The hinge kernel calculates the joint probability distribution of the angle combination of two hinged edge fragments. The hinge kernels are quantized into a 2D histogram to calculate the hinge feature. The joint probability of the orientations $\alpha$ and $\beta$ ($\alpha < \beta$) is quantized into a 2D histogram, resulting in a feature vector of dimension 253.

In order to build more powerful features, the joint feature distribution principle (JFD) was proposed by [115]. Following this principle, new features can be created by taking the joint distribution of features on adjacent positions or the joint distribution of different features in the same location. The Hinge feature was extended following the JFD, to create two new features, CoHinge and QuadHinge [113]. These new features are based on the spatial co-occurrence of the hinge. CoHinge is the joint distribution of the Hinge kernel on two different points $x_i$ and $x_j$ with Manhattan distance $l$ on the contours as in the following equation:

$$\text{CoHinge}(x_i, x_j) = [\text{Hinge}(x_i), \text{Hinge}(x_j)] \quad (5.1)$$

As each Hinge kernel has an alpha and beta value, CoHinge has two of each. One for every hinge kernel, which can be quantized into a 4D histogram.

QuadHinge incorporates curvature information of the contour fragments in the Hinge kernel by computing a fragment’s curvature measurement $C(F_c)$ for the contour fragments. Delta-Hinge is a rotation invariant feature that was proposed by
The feature is calculated from a feature network, with the differential operator between Hinge kernels as the kernel function $K^1$ defined as:

$$
\begin{align}
\Delta^n a(x_i) &= \frac{\Delta^{n-1} a(x_i) - \Delta^{n-1} a(x_i+\delta l)}{\delta l} \\
\Delta^n b(x_i) &= \frac{\Delta^{n-1} b(x_i) - \Delta^{n-1} b(x_i+\delta l)}{\delta l}
\end{align}
$$

QuillHinge is an extension of the quill-feature proposed by [28] that incorporates the Hinge kernel. It is the joint probability distribution $p(a, w)$ of the relationship between ink direction $a$ and the ink width $w$. This feature aims to capture information on the quill writing instrument. The QuillHinge feature is the probability of $p(a, \beta, w)$, which results in a 3D histogram. Lastly, we test the Triple chain code (TCC) feature proposed by [263]. The chain code of a pixel in character is one of the eight directions, where the next pixel is, denoted as 1 to 8. The TCC is defined as follows:

$$
TCC(x_i, x_{i+l}, x_{i+2l}) = [CC(x_i), CC(x_{i+l}), CC(x_{i+2l})]
$$

where $CC(x_i) \in 1, 2, ..., 8$ is the chain code value on position $x_i$ and $l$ is the Manhattan distance along the writing contours. Co-chain-code is an experimental extension of TCC, that is not published, but also used.

### 5.2.4.2 Grapheme-based method

In this study, the COnnected-COmponent COntrours (CO$^3$) method [254] is used as the grapheme-based method. The CO$^3$ is the contour obtained from each connected component in the image. In Figure 5.3, examples of this extraction can be seen. This shows several different extractions of the same Hebrew character. The images of the segmented graphemes are normalized to 50x50, as equal-sized input is necessary for the codebook.

A grapheme-based method aims to extract the individual graphemes of the handwriting. To capture the handwriting style of a manuscript, a statistical distribution of the graphemes is made. One of the methods to calculate this distribution is by using a codebook following a bag-of-words framework. Using a distance measure to find the most similar element in the codebook for each grapheme and taking the normalized histogram of this, the distribution can be determined. This results in a feature vector that is the same size as the number of nodes in the codebook.

![Figure 5.3: Examples of extracted graphemes (Alef, Bet, and Shin).](image)
Algorithm 1 SOTM procedure

\begin{algorithm}
    \begin{algorithmic}
    \State $y \leftarrow 1$
    \State randomly initialize $D_1$
    \State train $D_1$ using input patterns $\Omega(t)$ by a standard SOM method
    \While{$t \leq 7$}
    \State $t \leftarrow t + 1$
    \State initialize $D_t$ using $D_{t-1}$
    \State train $D_t$ using $\Omega(t)$ by a standard SOM method
    \EndWhile
    \State output $D = D_1, D_2, ..., D_t, ..., D_7$
    \end{algorithmic}
\end{algorithm}

5.2.4.3 Training codebook

In order to train the codebook, an unsupervised clustering method is often used. Two of the common methods are the k-means clustering [104] and the Self-Organizing Map (SOM) [150]. As these methods are unsupervised, they do not consider the known temporal information of the input. By training a single codebook, the subtle changes in style between the time periods can get lost. As the goal is to capture writing style changes over time, a semi-supervised method that takes the known information into account would be more suitable. A codebook method can be used based on the Self-Organizing Time Map (SOTM) proposed by [247], for dating historical manuscripts. The SOTM method works by training a sub-codebook $D_t$ for every time period $y(t)$.

The time periods are defined as follows:

$y(t) \in \{\text{Archaic, early-Hasmonean, Hasmonean, late-Hasmonean, early-Herodian, Herodian, late-Herodian}\}$

The initial sub-codebook $D_1$ is randomly initialized and trained using a SOM and only characters from $y(1)$, the Archaic time period. Then, sub-sequential codebooks are trained using the previous codebook $D_{t-1}$ as initialization for the SOM and characters from the time period in $y(t)$ as training data. The final codebook is the combination of all the sub-codebooks:

$D = \{D_1, D_2, ..., D_t, ..., D_7\}$

Algorithm 1 shows the pseudo-code for this procedure inspired by the work of [109]. In order to determine the feature vector for a document, a histogram is built by mapping each extracted grapheme to the most similar element in the codebook using the Euclidean distance measure. This is then normalized to produce the feature vector of a document, that can be used for further analysis. In Figure 5.4, examples of sub-codebooks for Hasmonean and Herodian are presented, showing visible changes in the writing style of the characters over time.
5.2.5  *Estimating time periods*

The final step of the model is to determine the time period using the calculated feature vector. The dating of a manuscript can be seen as either a classification into a time period or a regression to find a year estimate. Regression makes the most sense to use when the documents are written over a continuous period. This means there are no clear extended breaks, in which no manuscripts were written. This is the case for the DSS, as they are written over a continuous period. To do regression there need to be numerical year estimates on the labeled documents. For the DSS, these are only available on the 14C-dated documents. The scholar-labeled documents only have a time period estimate available. To train regression in this case, a year estimate needs to be determined for every document based on its time period. A simple solution is to take the center year of the time period. This holds an inherent error, as the true year can lie above or below the center within the range of the time period. The larger the time spans of the time periods, the larger this error becomes. When it is too large, classification is a better option, as this only aims to put the document in the correct time period, accepting this error inherently.

We have decided to do regression, as the time spans are small enough for the error to be not too large. The time period $y(t)$ has the corresponding (approximate) center year:

$c(t) \in \{-200, -130, -100, -55, -20, 15, 40\}$

(negative dates are BCE, positives are CE).

To do the regression, Support Vector Regression (SVR) [73], with a radial basis kernel, is trained using cross-validation and the labeled documents, with the estimated year as a label. This trained model can now be used to predict a manuscript’s date.

Figure 5.4: *Left:* A sub-codebook trained with Hasmonean characters; *Right:* A sub-codebook trained with Herodian characters.
5.3 EXPERIMENTAL RESULTS

In this section, the experimental procedures and the results from different approaches are presented. Each textural method and the grapheme method are evaluated. The grapheme method is trained on the manuscripts referenced by these Q-numbers. For each manuscript, graphemes were extracted from images that belong to it and were used to generate the histogram based on the codebook. The codebook itself is trained by taking all characters extracted from these labeled documents and training the sub-codebooks using the characters from its time period. For the textural methods, the same labeled Q-numbers are used. In Table 5.1, the number of labeled Q-numbers for each time period and the corresponding amount of images can be seen with their prior probabilities and the number of CO3 used. For the grapheme method, the feature vector is determined using the characters and the codebook for each document. Different sub-codebook sizes have been evaluated. For the textural methods, the feature vectors are calculated on every image belonging to the labeled Q-numbers. These are used to train an SVR model. The model is evaluated using 10-fold cross-validation.

Table 5.1: Prior probability, number of images, and number of graphemes (CO3) for each time period used in this experiment.

<table>
<thead>
<tr>
<th>Time period</th>
<th>Qnr</th>
<th>Prior</th>
<th>Image</th>
<th>N_{CO3}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Archaic</td>
<td>2</td>
<td>0.0101</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>Early-Hasmonean</td>
<td>29</td>
<td>0.1496</td>
<td>89</td>
<td>620</td>
</tr>
<tr>
<td>Hasmonean</td>
<td>46</td>
<td>0.1563</td>
<td>93</td>
<td>554</td>
</tr>
<tr>
<td>Late-Hasmonean</td>
<td>89</td>
<td>0.2050</td>
<td>122</td>
<td>1387</td>
</tr>
<tr>
<td>Early-Herodian</td>
<td>76</td>
<td>0.2555</td>
<td>152</td>
<td>2145</td>
</tr>
<tr>
<td>Herodian</td>
<td>8</td>
<td>0.1294</td>
<td>77</td>
<td>84</td>
</tr>
<tr>
<td>Late-Herodian</td>
<td>19</td>
<td>0.0941</td>
<td>56</td>
<td>974</td>
</tr>
</tbody>
</table>

5.3.1 Measures

To evaluate the SVR, two common performance evaluation methods for dating are used: the Mean Absolute Error (MAE) and the Cumulative Score (CS). The MAE is defined as follows:

\[
MAE = \frac{\sum_{i=1}^{N} |G(y_i) - P(y_i)|}{N}
\] (5.4)
Here \( y_i \) is the document and \( G(y_i) \) is the ground truth year estimate of the document, \( P(y_i) \) the predicted year estimate and \( N \) is the number of test documents. The CS method used is defined as follows per [89]:

\[
CS = \frac{N_{e\leq\alpha}}{N} \times 100\% \tag{5.5}
\]

Here \( N \) is the number of test documents and \( N_{e\leq\alpha} \) is the documents where the absolute error, \( e \), is below the acceptance threshold \( \alpha \). The CS method can be seen as giving the accuracy of the estimator at the acceptance threshold rate. The CS is a percentage score. The closer it is to 100\%, the better.

### 5.3.2 Sub-codebook size

A set of six different sub-codebook sizes has been analyzed using the aforementioned measurements. The sub-codebook size is the number of nodes \( n_{row} \times n_{col} \) used in each individual sub-codebook. The full codebook size is the combined size of all sub-codebooks. The tested sub-codebook sizes are: \( N_{sub} \in \{25, 100, 225, 400, 625, 900\} \).

The MAE in relation to the sub-codebook size can be seen in Figure 5.5. An increase in the sub-codebook size decreases the MAE until the size is 225. Then the MAE starts to go up again with larger standard deviations. Codebook size 225 performs the best with an MAE of 23.4 years. The CS(\( \alpha = 25 \)) in relation to the sub-codebook size can be seen in Figure 5.6. The graph shows that the CS(\( \alpha = 25 \)) improves with an increase in the sub-codebook size. The increase is marginal after a size of 100. For further graphs comparing the codebook with the textural methods, the sub-codebook size 225 (15x15) will be used as it has the best trade-off between MAE and CS(\( \alpha = 25 \)).

### 5.3.3 Overall results

In this section, we look at the overall results between the textural methods and the grapheme (codebook) method with a sub-codebook size of 225. For each method, the MAE, CS(\( \alpha = 1 \)) and CS(\( \alpha = 25 \)) have been determined. These are shown in Table 5.2. The codebook method performs the best by a large margin. It has a MAE of 23.4, CS(\( \alpha = 1 \)) of 19.4 and a CS(\( \alpha = 25 \)) of 60.6. These scores are far larger than the second-best method and the best textural method, QuadHinge.
Figure 5.5: Mean absolute error (in years) for varying sub-codebook sizes. Error bars represent the standard deviation between folds.

Figure 5.6: Mean cumulative score with \( \alpha = 25 \) for varying sub-codebook sizes. Error bars represent the standard deviation between folds.

5.3.4 Cumulative scores

Finally, CS with alpha rates, 1, 25, 50, 75, and 100 are tested for the codebook method and the best-performing textural method. A graph of this is shown in Figure 5.7. This shows that the codebook is always ahead of the textural method, but with larger acceptance rates, their performance levels become closer. They both have similar error rates, for every point on the graph. Additionally, for a visual representation of the system’s output, density plots of predicted dates and real dates are shown in Figure 5.8.

5.4 Discussion

Firstly, the aim of this study was to find out if applying a handwriting pattern analysis-based approach for dating the DSS can achieve good results. The results show that the grapheme-based method using a self-organizing time map as a codebook outperforms other textural methods. Among these, QuillHinge was designed for the manuscripts using a quill as the writing device, which was not used back when the DSS were written. This explains why QuillHinge is the lowest performing
Table 5.2: Results for textural methods and codebook (CO$^3$) with $n_{sub} = 225$.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
<th>CS($\alpha = 1$)</th>
<th>CS($\alpha = 25$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hinge</td>
<td>43.1 ± 6.4</td>
<td>0.5 ± 0.8</td>
<td>35.5 ± 5.7</td>
</tr>
<tr>
<td>CoHinge</td>
<td>42.5 ± 6.9</td>
<td>1.5 ± 1.4</td>
<td>37.0 ± 9.9</td>
</tr>
<tr>
<td>Delta-Hinge</td>
<td>44.3 ± 5.3</td>
<td>0.7 ± 1.1</td>
<td>35.3 ± 7.4</td>
</tr>
<tr>
<td>QuillHinge</td>
<td>55.4 ± 9.4</td>
<td>0.7 ± 1.1</td>
<td>23.5 ± 6.6</td>
</tr>
<tr>
<td>QuadHinge</td>
<td>42.4 ± 7.4</td>
<td>1.7 ± 0.8</td>
<td>37.5 ± 9.1</td>
</tr>
<tr>
<td>TCC</td>
<td>44.7 ± 6.8</td>
<td>1.0 ± 1.7</td>
<td>33.5 ± 5.8</td>
</tr>
<tr>
<td>CCC</td>
<td>42.3 ± 6.8</td>
<td>1.2 ± 1.1</td>
<td>35.0 ± 7.9</td>
</tr>
<tr>
<td>CO$^3$</td>
<td>23.4 ± 6.6</td>
<td>19.4 ± 9</td>
<td>60.6 ± 9.4</td>
</tr>
</tbody>
</table>

method and also gives clues about the writing implement, which is likely to be blunt. This is coherent with the structure of the characters and the idea of using tools similar to reed pens. In general, the reed pens are stiffer than the quills, and they do not retain a sharp point for a long time.

In order to explain the performance of the other methods, different aspects need to be considered. Any feature extraction method’s performance can be affected by two factors: scale and rotation. In the DSS, the handwriting forms can vary significantly in terms of their scale and rotation among fragments. For example, the fragments in Figure 5.1 and 5.2 have different character-shape angles in reference to the horizontal axis. The size of the handwriting can also differ among images. These can possibly influence performance measures. DeltaHinge is the only textural method that has rotation invariance. However, it does not show that this helps its performance in this application. This might suggest that a small rotation of the patterns does not affect the performance to a large degree for the DSS. As none of the methods is scale-invariant, the scale differences can still be a negative factor. For the grapheme-based method, the extracted graphemes are normalized and matched with the codebook. As it uses a similarity measure to match every grapheme with codebook nodes, the scale difference has a less significant impact. This could be one of the reasons for the grapheme-based method’s better performance.

An issue, not reflected directly in the results but important to note, is the imbalance of the labeled data. There is a low number of Archaic manuscripts in comparison to the other time periods. Because of the way SVR works, this can result in the system performing worse when predicting the date for a manuscript that is Archaic. In similar studies on different datasets, the time periods have a 25-year margin between each period and are called key years. The time periods for the DSS have margins in the range of 25-70 years. As the dates for the labeled manuscripts are estimated using the center year of the time period they belong to, these estimates have an inherent error affecting the MAE and CS. For example, the MPS dataset has more
labeled data with higher quality. Because of these factors, the results are not directly comparable. The new $^{14}$C-dates of the ERC project will be useful for a more precise date estimation.

![Cumulative Score Performance](image)

Figure 5.7: Mean cumulative score performance with varying error levels. Error bars represent the standard deviation between the folds.

Additionally, it might be the case that using SVR is too rigid of a solution for the textural methods. A way to change this would be to create a hit list of the closest labeled manuscripts, using a distance measure. By assigning weights to the ranks of the hit list, a date can be predicted by a linear combination of the weights and the hit-list manuscript dates. This would be similar to the k-nearest neighbors approach. Lastly, new textural features could be developed specifically for ancient Hebrew script and manuscript dating, taking into account the characteristics of this script and common aspects of the script that change over time. Using this feature in combination with other proposed solutions to problems might result in a high-performing textural feature.

![Density Plot](image)

Figure 5.8: Density plot of predicted date and real date for 10x10,15x15, and 20x20 codebooks.
5.5 CONCLUSIONS

This chapter has shown that the grapheme-based method with a SOTM performs better than other methods for dating the DSS. Possible reasons for this have been discussed, and solutions have been proposed. This study gives a good initial overview of what works in regards to dating the DSS and which problems and challenges still remain. By taking care of the discussed problems and exploring the proposed methods, we believe that the performance of the textural and grapheme-based methods can be improved. This work will remain as a benchmark, and further work by integrating precise dates, i.e., the $^{14}$C-dates, will improve the robustness of a dating tool for the DSS using pattern recognition techniques in the following chapter.

ACKNOWLEDGEMENTS FOR CHAPTER 5

The authors (from the original article) would like to thank Mladen Popović (Principal investigator of the ERC project, Faculty of Theology and Religious Studies, University of Groningen) for his valuable input in categorizing the time periods for the images. This work of this chapter has been supported by an ERC Starting Grant of the European Research Council (EU Horizon 2020): The Hands that Wrote the Bible: Digital Palaeography and Scribal Culture of the DSS (HandsandBible # 640497).