

Learning the Character Inventories of Undeciphered Scripts Using Unsupervised Deep Clustering

Logan Born¹

loborn@sfu.ca

M. Willis Monroe²

willis.monroe@ubc.ca

Kathryn Kelley³

kathrynerin.kelley@unibo.it

Anoop Sarkar¹

anoop@cs.sfu.ca

¹Simon Fraser University
School of Computing Science

²University of British Columbia
Department of Philosophy

³Università di Bologna
Dipartimento di Filologia Classica e Italianistica

Abstract

A crucial step in deciphering a text is to identify what set of characters were used to write it. This requires grouping character tokens according to visual and contextual features, which can be challenging for human analysts when the number of tokens or underlying types is large. Prior work has shown that this process can be automated by clustering dense representations of character images, in a task which we call “script clustering”. In this work, we present novel architectures which exploit varying degrees of contextual and visual information to learn representations for use in script clustering. We evaluate on a range of modern and ancient scripts, and find that our models produce representations which are more effective for script recovery than the current state-of-the-art, despite using just 2% as many parameters. Our analysis fruitfully applies these models to assess hypotheses about the character inventory of the partially-deciphered proto-Elamite script.

1 Introduction

One of the first tasks in decipherment is to solve an instance of the token-to-type problem by recognizing which tokens represent the same underlying character, and thereby to construct a list of every character used in the script (cf. [Gelb and Whiting 1975](#)). Accurate character inventories are important for decipherment, as they indicate patterns of frequency and adjacency which can reveal information about the underlying message. However, it can be challenging for human annotators to determine which characters are truly distinct: tokens with different appearances can represent the same underlying character (such as English **t** and **P**), while visually-similar tokens can represent distinct characters (such as **t** and **ŕ** or **β** and **B**).

This work introduces novel, VAE-based techniques for learning the character inventory of an unknown script by clustering images of character

tokens. We show, through a range of experiments on deciphered and undeciphered scripts from modern and ancient corpora, how the complexity and number of characters in a script impact our models’ ability to learn the underlying character inventory. On the ancient Cypro-Greek syllabary, our models outperform the recent Sign2Vec architecture ([Corazza et al., 2022a](#)) despite using just ~2% as many parameters. We also apply these models to the undeciphered proto-Elamite script (PE; [Dahl 2019](#)), and find that they independently replicate expert intuitions about the underlying character inventory and suggest new relationships between signs which have not yet been noted in prior work.

2 Methodology

[Corazza et al. 2022a](#) and [Corazza et al. 2022b](#) outline a two-step procedure for learning the character inventory of an unknown script by clustering images of character tokens. They first train an unsupervised neural encoder to learn vector representations for images of the characters in question. After training, they cluster these representations: the resulting clusters serve as an estimate for the script’s underlying character inventory. The authors demonstrate good performance on the ancient Cypro-Greek script, and fruitfully apply this technique to the study of a related, undeciphered script called Cypro-Minoan.

Our work follows the same overall approach, and investigates how changes to the encoder architecture, data quality, and training process can affect the final clustering.

2.1 Motivation

Sign2Vec ([Corazza et al., 2022a](#)) uses the ResNet18 encoder ([He et al., 2016](#)), which is an 18-layer convolutional stack with residual connections. ResNet was originally developed for object detection and segmentation on the ImageNet ([Deng et al., 2009](#)) and COCO ([Lin et al., 2014](#)) datasets, which in-

clude photorealistic depictions of complex scenes. In this setting, very deep networks are necessary to capture the full range of visual information present in the input images (He et al., 2016). By contrast, images of written characters tend to be visually simple: they often read clearly in greyscale or black-and-white, and can generally be broken down into simple lines, curves, or wedges against a uniform background. In light of this, we hypothesize that the ResNet encoder may be significantly over-parameterized for the task of script clustering. This may lead to longer training times than necessary, more expensive compute costs, and increased risk of overfitting when applied to low-resource decipherment corpora.

Additionally, one of the tasks used to train the Sign2Vec encoder is a masked prediction task, where information about a character must be recovered given the representations of the characters to its immediate left and right. This provides the model with a very narrow context window, which is sufficient for the experiments in the original work (Corazza et al., 2022a), but which we hypothesize may hamper the model’s performance in settings where wider context is available.

2.2 Model Architectures

In light of these limitations, we propose to compare four architectures which reduce the size of the encoder relative to Sign2Vec and incorporate varying degrees of context.¹

VAE All of our models are built around a variational autoencoder (VAE; Kingma and Welling 2014) with a convolutional encoder and a deconvolutional decoder (Figure 1). This architecture uses three stacked convolutional layers to learn vector representations $\mu, \sigma \in \mathbb{R}^d$ from an input image $\mathbf{x} \in \mathbb{R}^{n \times n}$. These are used to sample a “code” $\mathbf{z} \sim \mathcal{N}(\mu, \sigma)$. A stack of transposed convolutional layers decodes \mathbf{z} to an image $\tilde{\mathbf{x}} \in \mathbb{R}^{n \times n}$. This model is trained to minimize the reconstruction error of $\tilde{\mathbf{x}}$ with respect to the input \mathbf{x} :

$$\mathcal{L} = \text{BCE}(\tilde{x}, x) \quad (1)$$

where BCE is binary cross-entropy.

VAE+Neighbors Our second model adds an auxiliary masked prediction task (Figure 2). Let \mathbf{z}_{i-1} and \mathbf{z}_{i+1} be the encodings of the images to the

¹Code to be released at <https://github.com/MrLogarithm/cawl-clustering>

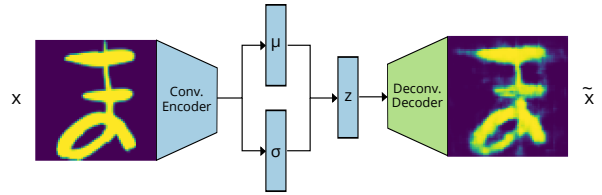


Figure 1: VAE architecture. This model reconstructs its input from a dense vector encoding.

direct left and right of a token \mathbf{x}_i . We learn a projection $M \in \mathbb{R}^{2d \times d}$ and decode $M(\mathbf{z}_{i-1} \oplus \mathbf{z}_{i+1})$ to produce an image $\tilde{\mathbf{x}}'_i$. We add a new loss term to Equation (1) to minimize the reconstruction error of $\tilde{\mathbf{x}}'_i$ with respect to \mathbf{x}_i :

$$\mathcal{L}_{\text{Neighbor}} = \text{BCE}(\tilde{\mathbf{x}}'_i, \mathbf{x}_i)$$

This is similar to the auxiliary task in Sign2Vec (Corazza et al., 2022a), with the difference that our model *draws* the masked sign, whereas Sign2Vec was trained to predict a property called its “pseudolabel” (see Section 2.3).

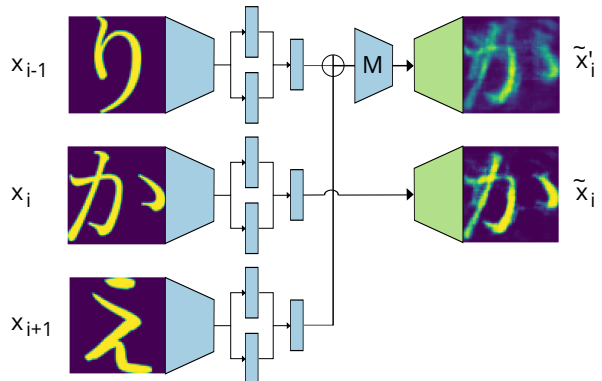


Figure 2: VAE+Neighbor architecture. This model adds the auxiliary task of reconstructing a character image given the encodings of the adjacent characters.

VAE+LSTM To include wider context, we propose a third architecture incorporating an autoregressive LSTM (Hochreiter and Schmidhuber, 1997) language model. The input to this model is a sequence of character images $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$. We encode each image using convolutional encoders with tied parameters to produce a sequence of codes $\{\mathbf{z}_1, \dots, \mathbf{z}_n\}$, and decode these using tied decoders to produce a sequence of images $\{\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_n\}$. Up to this point, the model is equivalent to a batched version of the basic VAE model. To incorporate context, we pass $\{\mathbf{z}_1, \dots, \mathbf{z}_n\}$ to a unidirectional LSTM, and use our VAE decoder to decode

the LSTM outputs to a second image sequence $\{\tilde{\mathbf{x}}'_1, \dots, \tilde{\mathbf{x}}'_n\}$.

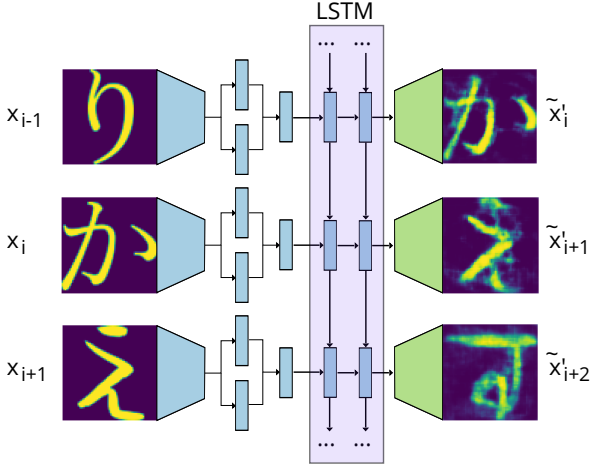


Figure 3: VAE+LSTM architecture. This model adds the auxiliary task of drawing the next token given a sequence of encodings for the preceding tokens.

We add the following loss term to Equation (1) to minimize the reconstruction error of this image sequence:

$$\mathcal{L}_{\text{LSTM}} = \sum_{i=1}^{n-1} \text{BCE}(\tilde{\mathbf{x}}'_i, \mathbf{x}_{i+1})$$

This can be viewed as an autoregressive character-level language modeling objective, where we wish to draw the image of the next character \mathbf{x}_{i+1} given all of the preceding characters $\mathbf{x}_1, \dots, \mathbf{x}_i$.

VAE+Transformer Our final model replaces the LSTM component from the previous model with a Transformer encoder stack (Vaswani et al., 2017); we obtain the output image sequence $\{\tilde{\mathbf{x}}'_1, \dots, \tilde{\mathbf{x}}'_n\}$ by decoding the top layer of this Transformer. We train this model on a masked language modeling task: we mask input tokens at random by replacing their images with standard Gaussian noise, and train the model to recover the unmasked image sequence by adding the following term to Equation (1):

$$\mathcal{L}_{\text{Transformer}} = \sum_{i=1}^n \text{BCE}(\tilde{\mathbf{x}}'_i, \mathbf{x}_i)$$

2.3 Training Details

At training time, we use a denoising technique (Vincent et al., 2008, 2010) to regularize

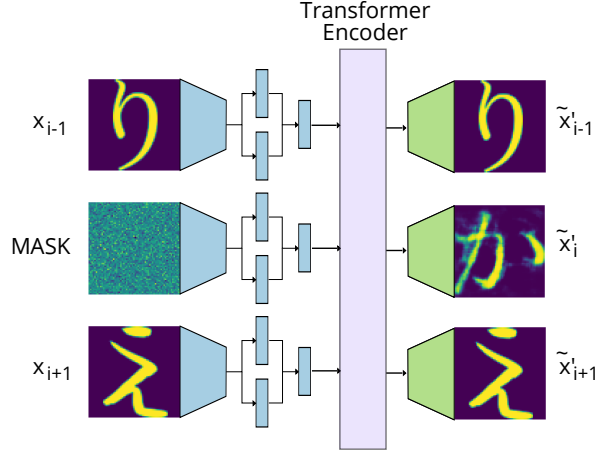


Figure 4: VAE+Transformer architecture. This model adds the auxiliary task of reconstructing characters which have been masked by random Gaussian noise.

our models: we apply a random transformation (rotation of up to 45 degrees, shear of up to 25 degrees, and a random scale factor between 0.4 and 1) to each input image, while keeping the target of the reconstruction loss unchanged.

All of our models are trained using stochastic gradient descent (SGD) to minimize Eq. (1), plus the appropriate model-specific auxiliary loss ($\mathcal{L}_{\text{Neighbor}}$, $\mathcal{L}_{\text{LSTM}}$, or $\mathcal{L}_{\text{Transformer}}$), plus a *pseudolabel* loss term \mathcal{L}_{Ψ} which we describe below. We jointly minimize the sum of all of the relevant loss terms in a single pass, with no pretraining and no warmup steps.

Pseudolabels We follow Corazza et al. 2022a in using a soft, unsupervised technique to organize our models’ encodings into loose clusters. This technique, inspired by the neural clustering algorithm DeepCluster-v2 (Caron et al., 2018), begins by clustering the encodings using K-Means with an **arbitrary** number of clusters k . Let C be a matrix with columns corresponding to the K-Means centroids (normalized to unit length). Let \mathbf{z}_i be an arbitrary encoding, let C_j be the centroid which is closest to \mathbf{z}_i , and let \mathbf{y}_i be a one-hot vector with a one in the j th position. The pseudolabel loss is then given by:

$$\mathcal{L}_{\Psi} = \sum_{i=1}^n \text{CCE}\left(\frac{\mathbf{z}_i}{\|\mathbf{z}_i\|} C, \mathbf{y}_i\right) \quad (2)$$

where CCE is categorical cross-entropy. For each \mathbf{z}_i , this constructs a probability distribution $\frac{\mathbf{z}_i}{\|\mathbf{z}_i\|} C$ over k categories, where the mass in each category is proportional to the similarity between \mathbf{z}_i and the

corresponding centroid. Minimizing this loss concentrates the mass of the distribution into a single term; in other words, this performs a soft clustering by pulling each z_i towards the nearest centroid in the embedding space.

We follow [Corazza et al. 2022a](#) in using a pseudolabel loss with 100 centroids, which we recompute using K-Means at the beginning of each training epoch.

3 Data

We aim to apply these models to the study of the undeciphered proto-Elamite script, which is attested across ca. 1581 clay tablets recovered from the ancient city of Susa and other parts of the Iranian plateau. These texts have been transliterated by domain experts using a work-in-progress list of about 1500 distinct signs;² the most complete and up-to-date transliterations are hosted by the Cuneiform Digital Library Initiative³ and described in a recent survey by [J. Dahl \(2019\)](#). Each sign has an accompanying digital image, also produced by Dahl, depicting its “archetypal” form. These images smooth over many of the irregularities of the original shapes drawn on clay, while still preserving slight visual differences between tokens which may actually represent the same underlying character (such as 𐎧 and 𐎧). They therefore represent an intermediate level of detail that is cleaner than segmented images of the original texts, yet still faithful to the original hand. We convert the entire transliterated proto-Elamite corpus⁴ into a set of image sequences by replacing each transliterated sign name with the corresponding sign image (Figure 5). Table 1 summarizes the token count for the resulting dataset.

We evaluate our models on their ability to recover three scripts whose character inventories are already known (English, Japanese, and Cypro-Greek). We construct an English dataset by extracting the first 33k alphanumeric tokens (approximately the same number of tokens as are attested in

²Different sign-counting methodologies can yield sign counts as low as 287 or as high as 1623 ([Born et al., 2019](#)), depending on whether numerals, tilde-variants, complex graphemes, and other categories of grapheme are included. To further complicate matters, the signlist continues to vary as transliterations are updated and sign names are revised. At the time of publication, a list of sign names which are currently in use can be found at <https://cdli.mpiwg-berlin.mpg.de/resources/token-lists>

³<https://cdli.mpiwg-berlin.mpg.de/>

⁴<https://cdli.mpiwg-berlin.mpg.de/search?period=proto-elamite&genre=administrative>



Figure 5: Samples of image sequences from our PE (top), En (middle) and Jp (bottom) datasets.

Language	# Characters	#Tokens	# Images
EN	62	33k	3410
JP	938	33k	1607
CG	55	3k	3005
PE	—	35k	1319

Table 1: Size and character inventories of scripts used for training. PE is undeciphered, and the size of its character inventory remains unknown.

proto-Elamite) from the WikiText-2 corpus ([Merity et al., 2016](#)). We convert this text into image sequences by replacing each character token with a handwritten image of that character. We use images from [de Campos et al. 2009](#), who provide 55 handwritten instances of all 62 upper- and lowercase English letters and digits: one of these 55 images is chosen at random each time a character occurs. The resulting sequences (Figure 5) imitate the level of detail in our proto-Elamite data, in that each letter is attested by multiple distinct images, and the same image can be used for multiple tokens.

We construct a Japanese dataset according to the same procedure, using the first 33k tokens from the Japanese Tatoeba corpus ([Artetxe and Schwenk, 2018](#); [Tiedemann, 2012](#)). As we do not have handwritten character images for Japanese, we instead extract the glyphs from two Japanese fonts (Yuji Boku and Zen Old Mincho). The Japanese writing system uses three scripts: *kanji* which are highly logographic, and two syllabaries called *hiragana* and *katakana*. Similarly, proto-Elamite has been speculated to contain a set of possibly-syllabic signs, together with a large number of logograms ([Dahl, 2019](#)). Syllabic signs convey phonetic information that can provide crucial insights for decipherment, and are therefore a major focus of decipherment efforts on this script. In our Japanese experiments (see Section 4), we therefore train on the entire script, but only evaluate the model’s ability to recover the two syllabaries.

We use the same Cypro-Greek data as [Corazza et al. 2022a](#). Unlike the other datasets, this uses manually-segmented images from hand-drawn

copies of artifacts bearing the Cypro-Greek script. There is therefore a greater degree of variation between the character shapes in this data, and no two tokens ever have identical images. This means that our three datasets fall along a cline from fully naturalistic, handwritten sequences (Cypro-Greek), to synthetic sequences derived from handwritten images (English), to synthetic sequences derived from digital fonts (Japanese).

We trim extraneous whitespace from all character images and resize them to 64×64 pixels with a single grayscale color channel before training.⁵

4 Experiments

We train each of the models from Section 2.2 on the four corpora detailed above (see Appendix B for hyperparameters and additional training details). After training, we encode each image using the trained encoder and cluster the resulting encodings using (i) agglomerative clustering with varying numbers of clusters (English, Japanese, proto-Elamite) or (ii) DBSCAN (Ester et al., 1996) with varying ϵ (Cypro-Greek). We use DBSCAN for Cypro-Greek to enable a fair comparison against the results in Corazza et al. 2022a; however, we find that DBSCAN is generally not effective when clustering the other scripts. When clustering with DBSCAN, we follow Corazza et al. 2022a in using a minimum cluster size of 2, to imitate a decipherment setting where the true number and frequency of characters is unknown; for the other scripts we vary the number of clusters for the same reason.⁶

For English, Japanese, and Cypro-Greek, we report homogeneity, completeness, and V-measure (Rosenberg and Hirschberg, 2007) relative to the gold labels. Homogeneity ranges from 0 to 1, where 1 means that each cluster contains instances from just one of the underlying characters, and smaller values imply that some clusters combine instances of two or more distinct characters. Similarly, a completeness of 1 means that each of the underlying characters is represented by a single

⁵Rescaling the images to a fixed size obscures the height of the original character (see Fig. 5, where って is indistinguishable from つて). For this reason, our Japanese evaluation only tests the model’s ability to recover the gojūon, dakuon, and handakuon, ignoring the yōon, sokuon, and small vowels which are distinguished only by size.

⁶The present work will otherwise ignore the problem of selecting the correct number of clusters, for which a variety of heuristics have been proposed in prior work (Rousseeuw 1987; Thorndike 1953; Sugar and James 2003; Honarkhah and Caers 2010; Tibshirani et al. 2002 i.a.).

cluster, while smaller values mean that some characters have been divided across multiple clusters. Intuitively, low homogeneity scores mean that a clustering merges together characters which are actually distinct, while low completeness means that it splits some characters into subgroups that are not underlyingly distinct. V-Measure is the harmonic mean of homogeneity and completeness.

DBSCAN can label samples as outliers (and thus, not part of any cluster): we only evaluate on tokens which it assigns to a cluster.

In our Japanese experiments, we only evaluate on hiragana and katakana, in imitation of the proto-Elamite setting where we eventually aim to understand the divisions of a putative syllabary comprising only a subset of the overall script.

In our Cypro-Greek experiments, we compare against the Sign2Vec and DeepCluster-v2 results reported in Corazza et al. 2022a. For the other scripts, we compare against agglomerative clusterings over the input images.

In proto-Elamite, where the ground truth is not known, we perform a qualitative evaluation in collaboration with domain experts. We look for sets of tokens which are assigned to the same cluster by our VAE+Neighbor, VAE+LSTM, or VAE+Transformer model, but belong to *different* clusters in the vanilla VAE model. The vanilla VAE differs from the other models in that it lacks contextual information; therefore, any groupings which are absent from this model’s output likely reflect primarily contextual similarities. Contextual resemblances are harder for human annotators to notice than visual resemblances, and so we expect these groupings to reflect similarities which may have been overlooked in prior work. We collaborate with domain experts to assess how these token groupings relate to their intuitions about this script.

5 Results

5.1 Modern Scripts

Figure 6 plots V-Measure from agglomerative clusterings over our models’ representations of handwritten English letters (Appendix A shows the breakdown into homogeneity and completeness scores). The curve for the baseline is obtained by clustering the raw character images, rather than their encodings.

All four of our proposed models are able to recover the underlying script more accurately than the baseline. When the number of clusters is close

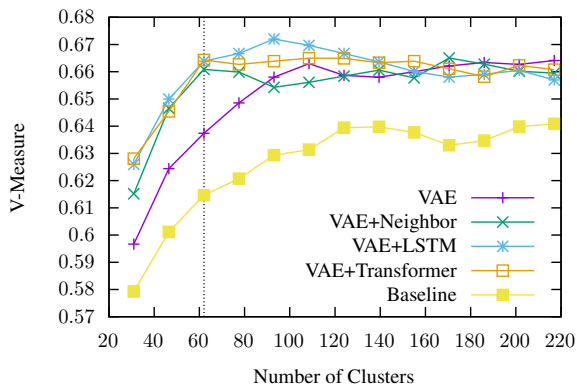


Figure 6: V-Measure on handwritten English. The true character inventory comprises 52 upper- and lowercase letters plus 10 digits.

to the true size of the alphabet, our LSTM and Transformer-based models achieve the highest performance, which supports our hypothesis that wider context windows allow for more accurate script recovery.

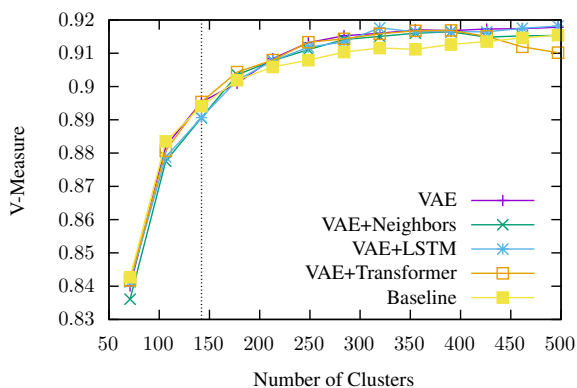


Figure 7: V-Measure on a synthetic mixture of Japanese fonts. The target character inventory comprises 142 hiragana and katakana (46 gojūon, 20 dakuon, and 5 handakuon each).

Figure 7 plots the same metrics for our synthetic Japanese dataset. In this setting, the differences between models are much less pronounced: the context-aware models do not exhibit the same advantage as in English, and in fact the VAE+LSTM model fails to outperform the naive baseline when the number of clusters exactly matches the true number of underlying characters. When the number of clusters is much larger than the true number of signs, our models do outperform the baseline, however, the contextual models continue to slightly underperform the contextless VAE on average. Regardless of the number of clusters chosen, the V-Measure for Japanese is always much higher than for English.

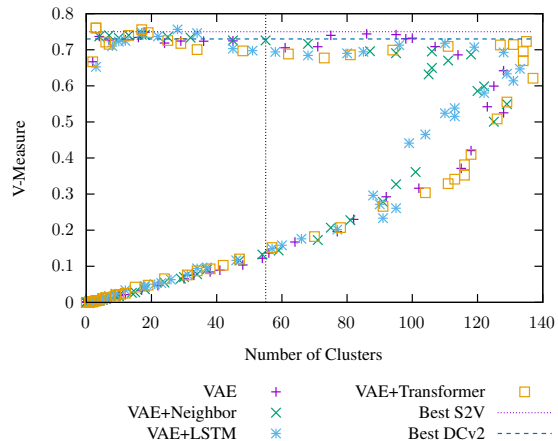


Figure 8: V-measure vs number of clusters for DBSCAN clusterings on Cypro-Greek. We evaluate over the interval $0.1 \leq \varepsilon < 8$ in steps of 0.1. The dotted line represents the true number of signs in the script.

These differences between English and Japanese are likely due, in part, to the fact that there are only two distinct images per character in the Japanese data, compared to 55 in English. The Japanese data are also fully synthetic, whereas the English is handwritten. This may make the Japanese task too easy (despite covering a much larger number of unique characters) to the point that contextual models are not needed. This is nevertheless a useful result, as it suggests that the difficulty of script clustering depends less on the number of graphemes than on the degree of variation between allographs.

5.2 Cypro-Greek

Table 2 compares the best result from each of our models against the best results reported in Corazza et al. 2022a; Figure 8 shows our full results for different values of DBSCAN’s ε parameter. Our best models (VAE+LSTM and VAE+Transformer) outperform the Sign2Vec baseline, and all of our models outperform DeepCluster-v2 (Caron et al., 2018) which was the inspiration for Sign2Vec. Although our V-measure gains are modest, we emphasize that our models use **~98% fewer parameters than Sign2Vec**, and **~99% fewer than DeepCluster-v2**. This supports our hypothesis that the ResNet encoder is over-parameterized for the task of script clustering, and demonstrates that accurate script recovery is clearly possible even with much more lightweight architectures.

Figure 9 plots homogeneity and completeness vs number of clusters for each of our CG models. Each model exhibits a unique trend: the

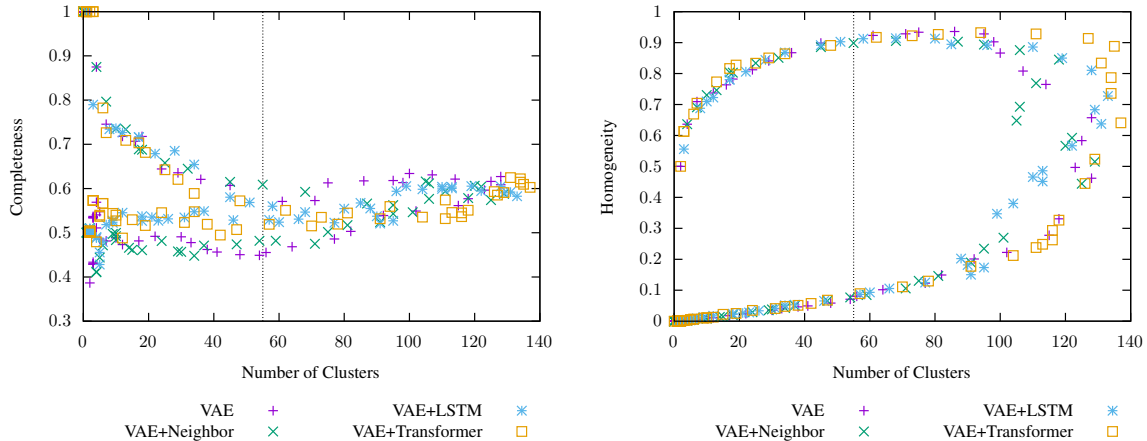


Figure 9: Completeness (left) and homogeneity (right) vs number of clusters for DBSCAN clusterings on Cypro-Greek. We evaluate over the interval $0.1 \leq \varepsilon < 8$ in steps of 0.1. The dotted line represents the true number of signs in the script.

	$V \uparrow$	Parameters \downarrow
DeepCluster-v2 (Corazza et al., 2022a)	0.73	> 23M
Sign2Vec (Corazza et al., 2022a)	0.75	> 11M
VAE (Ours)	0.75	0.215M
VAE+Neighbor (Ours)	0.74	0.215M
VAE+LSTM (Ours)	0.76	0.218M
VAE+Transformer (Ours)	0.76	0.227M





Table 2: V-measure (V) and parameter counts for Cypro-Greek. Best results for each model from Figure 8 and Corazza et al. 2022a.




VAE+Transformer exhibits less variation in its completeness scores across a range of clustering sizes, while the other models exhibit a more pronounced fall and rise as the number of clusters increases. All models are capable of achieving comparable homogeneity in the neighborhood surrounding the true number of clusters, but the VAE+Transformer maintains high homogeneity up to a much higher number of clusters than the other models. Geshkovski et al. 2023 have argued that self-attention mechanisms cause tokens to cluster around certain attracting states in the representation space; pseudolabeling (Caron et al., 2018) is intended to have the same effect. We speculate that the VAE+Transformer’s strong performance may result in part from these behaviours reinforcing one another to perform a more effective soft clustering at training time.

5.3 Proto-Elamite

Table 3 shows pairs and triplets of proto-Elamite characters which have distinct labels in the working signlist and VAE clustering, yet occupy the

same cluster in the VAE+Neighbor, VAE+LSTM, or VAE+Transformer clusterings.⁷

The VAE+Neighbor model differs from the working signlist and VAE clustering in only two places, merging M332~g  with M297~B  and M356~B  with M327~N . Neither merger appears to reflect any known similarities in how these signs are used.

By contrast, the 6 mergers proposed by the VAE+LSTM model appear much more plausible. One cluster combines tokens which are currently labeled as M362  and M362~a , which adds hatching to the central circle in a manner resembling “gunuification” in early cuneiform. The ~ notation in the working sign name explicitly acknowledges that experts believe M362~a may⁸ be a graphic variant of M362; both signs have been glossed as ‘nanny goat’ (Dahl, 2005), and previous scholarship has already acknowledged a likely equivalence between M362~a and another hatched variant called M362~b (Dahl, 2005) .

The output from our VAE+Transformer differs the most from the working signlist, suggesting 17 sets of shapes which may represent the same

⁷For the VAE model, we use 1306 clusters, which equals the number of unique sign images available at the time we created our training data. We cluster the other models using $3.5 \times$ this many clusters; using such a large number helps to guarantee that the observed groupings reflect legitimate similarities and are not simply a side effect of compressing too many tokens into too few groups.

⁸Specifically, ~ followed by a number marks one sign as a graphic variant of another; ~ followed by a letter means that the sign *may* be a variant of another, but experts remain agnostic in the absence of further evidence.

VAE+Neighbor	VAE+LSTM	VAE+Transformer		

Table 3: Pairs/triplets of character images which have distinct labels in the working signlist, but which our models merge into single clusters.

underlying character. A significant number of these are *complex graphemes*, where one glyph has been drawn inside the whitespace at the center of another. Previous work has suggested that the meaning of a complex grapheme is principally determined by its outer component (Born et al., 2021), and indeed most of the merges proposed by the VAE+Transformer occur between complex graphemes with the same outer part. This suggests that the model has rediscovered the same pattern identified in prior work, and that these particular merges do reflect plausible groupings on the part of our model. Many of the other merges occur between signs which are already labeled as possible variants in the working signlist (such as M029~a and M029~b , or the possible syllables M387~h and M387~ca) and thus appear similarly plausible.

Of greater interest are cases such as M209~a , M210~f , and M195+M057 . The working signlist labels these as wholly distinct characters, and no explicit relationship between these signs has been proposed in prior work. However, the visual resemblance between M209~a and M210~f is undeniable; both signs occur in texts which contain the “yoke” character M054 , and both occur in texts which appear to record amounts of grain (M209~a appears with the speculative grain capacity sign M354 , while M210~f occurs with the more common container sign M288). Moreover, in one text M209~a is attested alongside a related variant of M210, labeled M210~d . Given these signs’ visual resemblance to a plant sprouting from a field, and their association with yokes and grain accounts, we believe it is reasonable for the model to have grouped these characters under the same umbrella. M195+M057 is also attested in texts alongside both M288 and M354; although it does not occur next to the yoke sign M054, it often occurs near the sign M003~b , which is speculated to be another field utensil and which experts note is

“related to M054” (Dahl, 2007). Both M195+M057 and M209~a are also attested as “headers”, the first sign of a document which is believed to offer global context for interpreting the following text (Damerow and Englund, 1989; Born et al., 2022). While we are skeptical that M195+M057 is truly the same underlying character as M209~a and M210~f, they clearly share contextual similarities and are attested in comparable, apparently agricultural, contexts. This demonstrates our proposed model’s ability to detect contextual parallels which are helpful for understanding the possible relationships between signs in this script.

By reducing the number of clusters to force additional merges, we can obtain yet more groupings of the sort reported in Table 3. For example, when we reduce the number of clusters in the VAE+Transformer model by a factor of $\frac{1}{7}$, a new merger appears between M175+M131~d and M157+M131~d . The outer components M157 and M175 differ only in the shape of the protrusion at the top of the box, and a merger between these signs is tentatively expected based on current understandings of the corpus. Other mergers which appear, and which are also expected based on current understandings of the corpus, include M056~f with M056~e , both signs being understood to depict a plow; M075~ff , M075~g , M075~h , and M075~o apparently depicting minor variations on a sprouting plant; and M111~c , M111~d , and M111~e which differ only in the direction of the internal hatching. Such cases serve as useful confirmations of experts’ current understanding of sign use in this script.

The cases reported so far represent only a small fraction of the candidate mergers which can be extracted from our models, and we are optimistic that this work will continue to give rise to useful insights as experts take the opportunity to investigate this space more fully.

6 Related Work

Scribal hand identification (Popović et al., 2021; Srivatsan et al., 2021) is a related task which seeks to cluster instances of characters from a known script according to the hand which wrote them.

Yin et al. (2019) describe a system for segmenting, transliterating, and deciphering images of historical manuscripts. In the transliteration step, their model implicitly learns an underlying script by clustering character representations obtained from a Siamese neural network trained to discriminate between characters from known scripts. This network learns character representations without access to context, similarly to our vanilla VAE and the DeepCluster-v2 baseline in Corazza et al. 2022a.

In a setting where the underlying script is already known, Dencker et al. (2020) and Gordin et al. (2020) also describe systems for automated transliteration from images of cuneiform text.

Kelley et al. (2022) study the character inventory of proto-Elamite using a model similar to our VAE+LSTM. Their model is not variational; it uses softmax decoding over a fixed vocabulary initialized to match the working signlist, which biases it towards recovering the same divisions speculated by experts. Our models use a deconvolutional decoder, which sidesteps this bias by allowing an open vocabulary. Their evaluation does not test on any known scripts, which makes it challenging to determine the accuracy of the clusters their model produces.

7 Conclusion

We have described four models which add varying degrees of contextual information to a VAE, and have shown how these can be used to cluster token images to recover a script’s character inventory. On the ancient Cypro-Greek script, our best models meet or outperform the state-of-the-art Sign2Vec baseline using just $\sim 2\%$ as many parameters, which supports our claim that written text lacks the visual complexity to warrant models of the depth used in other image processing applications. Our English and Cypro-Greek experiments also demonstrate that contextual models are more effective for script recovery than contextless models. On synthetic Japanese data, which contains many distinct graphemes but little variation between allographs, our models achieve extremely high V-Measure (>0.91), suggesting that they handle large character inventories more easily than

they handle allography.

We apply our models to study the undeciphered proto-Elamite script, and show that they capture existing intuitions about this script as well as suggest new parallels between signs which have never been noted in prior work. Our best insights for proto-Elamite come from the LSTM and Transformer models, while for Cypro-Greek our VAE+Neighbor model is the only one which produces a clustering with precisely the same number of clusters as there are signs in the underlying script. This indicates that it is useful to consider models with varying access to contextual information according to the number of long-distance contextual dependencies the input script is expected to exhibit.

Limitations

Proto-Elamite is undeciphered, which means that our results on this script cannot be compared to any known ground truth. We attempt to ground our results by situating them relative to current Assyriological scholarship instead.

Writing systems exhibit considerable variation in terms of the number of characters used, the visual complexity of those characters, and the degree to which they represent phonetic information. Although we try to cover a range of alphabetic and non-alphabetic scripts in our evaluations, we cannot cover all possible cases, and focus on those which have some similarity to the proto-Elamite script which is the main concern of our work.

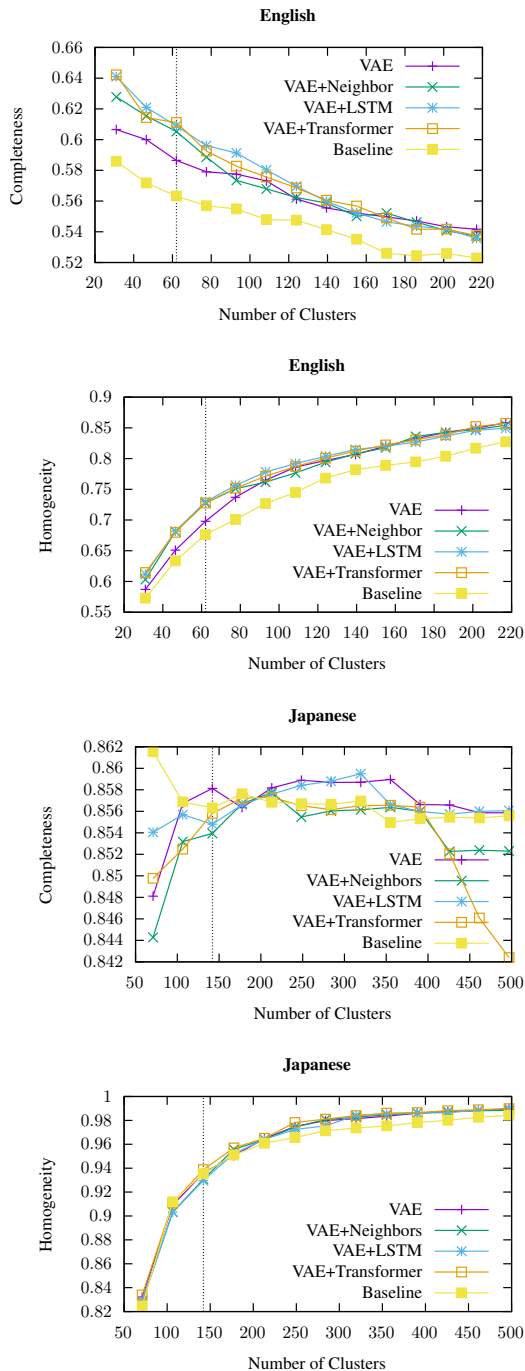
References

- Mikel Artetxe and Holger Schwenk. 2018. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *arXiv:1812.10464v2*.
- Logan Born, Kate Kelley, Nishant Kambhatla, Carolyn Chen, and Anoop Sarkar. 2019. **Sign clustering and topic extraction in Proto-Elamite**. In *Proceedings of the 3rd Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 122–132, Minneapolis, USA. Association for Computational Linguistics.
- Logan Born, Kathryn Kelley, M. Willis Monroe, and Anoop Sarkar. 2021. **Compositionality of complex graphemes in the undeciphered Proto-Elamite script using image and text embedding models**. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4136–4146, Online. Association for Computational Linguistics.

- Logan Born, M. Monroe, Kathryn Kelley, and Anoop Sarkar. 2022. [Sequence models for document structure identification in an undeciphered script](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9111–9121, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. [Deep clustering for unsupervised learning of visual features](#). In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV*, volume 11218 of *Lecture Notes in Computer Science*, pages 139–156. Springer.
- Michele Corazza, Fabio Tamburini, Miguel Valério, and Silvia Ferrara. 2022a. [Contextual unsupervised clustering of signs for ancient writing systems](#). In *Proceedings of the Second Workshop on Language Technologies for Historical and Ancient Languages*, pages 84–93, Marseille, France. European Language Resources Association.
- Michele Corazza, Fabio Tamburini, Miguel Valério, and Silvia Ferrara. 2022b. [Unsupervised deep learning supports reclassification of Bronze age cypriot writing system](#). *PLOS One*, 17(7):1–22.
- Jacob L. Dahl. 2005. Animal husbandry in Susa during the proto-Elamite period. *Studi Micenei ed Egeo-Anatolici*, 47:81–134.
- Jacob L. Dahl. 2007. Proto-Elamite signlist. Unpublished notes on the working signlist for proto-Elamite.
- Jacob L. Dahl. 2019. *Tablettes et fragments Proto-élamites / Proto-Elamite Tablets and Fragments*, volume XXXII of *Textes Cunéiformes du Louvre*. Éditions Khéops.
- Peter Damerow and Robert K. Englund. 1989. *The proto-Elamite texts from Tepe Yahya*, volume 39 of *American School of Prehistoric Research: Bulletin*. Peabody Museum, Cambridge, Massachusetts.
- Teófilo Emídio de Campos, Bodla Rakesh Babu, and Manik Varma. 2009. Character recognition in natural images. In *VISAPP 2009 - Proceedings of the Fourth International Conference on Computer Vision Theory and Applications, Lisboa, Portugal, February 5-8, 2009 - Volume 2*, pages 273–280. INSTICC Press.
- Tobias Dencker, Pablo Klinkisch, Stefan M. Maul, and Björn Ommer. 2020. [Deep learning of cuneiform sign detection with weak supervision using transliteration alignment](#). *PLOS One*.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. [ImageNet: A large-scale hierarchical image database](#). In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE.
- Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. [A density-based algorithm for discovering clusters in large spatial databases with noise](#). In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96*, page 226–231. AAAI Press.
- I. J. Gelb and R. M. Whiting. 1975. [Methods of decipherment](#). *Journal of the Royal Asiatic Society of Great Britain and Ireland*, (2):95–104.
- Borjan Geshkovski, Cyril Letrouit, Yury Polyanskiy, and Philippe Rigollet. 2023. [The emergence of clusters in self-attention dynamics](#).
- Shai Gordin, Gai Gutherz, Ariel Elazary, Avital Romich, Enrique Jiménez, Jonathan Berant, and Yoram Cohen. 2020. [Reading Akkadian cuneiform using natural language processing](#). *PLOS One*.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. [Deep residual learning for image recognition](#). In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. [Long short-term memory](#). *Neural Comput.*, 9(8):1735–1780.
- Mehrdad Honarkhah and Jef Caers. 2010. [Stochastic simulation of patterns using distance-based pattern modeling](#). *Mathematical Geosciences*, 42:487–517.
- Kathryn Kelley, Logan Born, M. Willis Monroe, and Anoop Sarkar. 2022. [Image-aware language modeling for proto-Elamite](#). *Lingue e linguaggio, Rivista semestrale*, (2/2022):261–294.
- Diederik P. Kingma and Jimmy Ba. 2015. [Adam: A method for stochastic optimization](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Diederik P. Kingma and Max Welling. 2014. [Auto-encoding variational Bayes](#). In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.
- Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. [Microsoft COCO: common objects in context](#). *CoRR*, abs/1405.0312.
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. [Pointer sentinel mixture models](#). *CoRR*, abs/1609.07843.
- Mladen Popović, Maruf A. Dhali, and Lambert Schomaker. 2021. [Artificial intelligence based writer identification generates new evidence for the unknown scribes of the Dead Sea scrolls exemplified by the great Isaiah scroll \(1QIsa^a\)](#). *PLOS ONE*, 16(4):1–28.

- Andrew Rosenberg and Julia Hirschberg. 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)*, pages 410–420, Prague, Czech Republic. Association for Computational Linguistics.
- Peter J. Rousseeuw. 1987. [Silhouettes: A graphical aid to the interpretation and validation of cluster analysis](#). *Journal of Computational and Applied Mathematics*, 20:53–65.
- Nikita Srivatsan, Jason Vega, Christina Skelton, and Taylor Berg-Kirkpatrick. 2021. [Neural representation learning for scribal hands of Linear B](#). In *Document Analysis and Recognition – ICDAR 2021 Workshops: Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II*, page 325–338, Berlin, Heidelberg. Springer-Verlag.
- Catherine A Sugar and Gareth M James. 2003. [Finding the number of clusters in a dataset](#). *Journal of the American Statistical Association*, 98(463):750–763.
- Robert L. Thorndike. 1953. [Who belongs in the family?](#) *Psychometrika*, 18:267–276.
- Robert Tibshirani, Guenther Walther, and Trevor Hastie. 2002. [Estimating the Number of Clusters in a Data Set Via the Gap Statistic](#). *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 63(2):411–423.
- Jörg Tiedemann. 2012. [Parallel data, tools and interfaces in OPUS](#). In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC’12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.
- Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. [Extracting and composing robust features with denoising autoencoders](#). In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 1096–1103. ACM.
- Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. [Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion](#). *J. Mach. Learn. Res.*, 11:3371–3408.
- Xusen Yin, Nada Aldarrab, Beáta Megyesi, and Kevin Knight. 2019. [Decipherment of historical manuscript images](#). In *2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, September 20-25, 2019*, pages 78–85. IEEE.

A Additional Figures



B Reproducibility Details

The code for our models will be published at <https://github.com/MrLogarithm/cawl-clustering>. All of the models in this work are implemented with PyTorch. All settings use an encoder with the following structure:

```
Sequential(
  Dropout(0.5)
  Conv2d(1 input channel, 3 output
```

```
channels, kernel size 8)
ReLU()
Conv2d(3 input channels, 6 output
channels, kernel size 8)
ReLU()
MaxPool2d(kernel size 3, stride
length 3)
Conv2d(6 input channels, 8 output
channels, kernel size 8)
ReLU()
MaxPool2d(kernel size 3, stride
length 3)
Flatten()
Dense(72 input dims, 16 output
dims)
)
```

)

A pair of Dense(16, 16) layers project the encoded output to μ and σ .

All settings use a decoder with the following structure:

```
Sequential(
  ConvTranspose2d(16 input channels,
60 output channels, kernel size
8)
BatchNorm2d(60 channels)
ReLU()
ConvTranspose2d(60 input channels,
30 output channels, kernel size
8, stride length 2)
BatchNorm2d(30 channels)
ReLU()
ConvTranspose2d(30 input channels,
15 output channels, kernel size
8, stride length 2)
BatchNorm2d(15 channels)
ReLU()
ConvTranspose2d(15 input channels,
1 output channel, kernel size
15, stride length 1)
Sigmoid()
)
```

)

The VAE+LSTM model uses a single-layer unidirectional LSTM with a hidden dimension of size 16. The VAE+Transformer uses a 6-layer TransformerEncoder with 8 heads per layer, input and output dimensions of size 16, and 0.5 dropout. We apply a standard sinusoidal positional encoding to the Transformer inputs following Vaswani et al. (2017).

In the VAE+LSTM and VAE+Transformer models, we re-apply the reparameterization trick from [Kingma and Welling 2014](#) to the LM outputs before decoding the image sequence. We add new Dense(16, 16) layers to compute μ and σ at this stage, separate from those used to compute μ and σ within the VAE itself. When computing the overall KL divergence loss for these models, we sum the KL divergence from these projections with that of the VAE projections.

We train on sequences of length 50 using the Adam optimizer ([Kingma and Ba, 2015](#)) with learning rate 0.001. When computing the loss, we scale the KL divergence loss term by 0.45. The LR and loss scaling hyperparameters were tuned via a small manual grid search. We recompute pseudolabel assignments at the start of every 600th training iteration.