

## S1 Appendix.

**Decoding across HMM latents: fitting and evaluation.** Consider two HMMs  $u$  and  $v$ , of sizes  $M(u)$  and  $M(v)$ , both candidate models of a dataset  $\mathcal{X}$ . Following (??), each HMM can be used to infer latents from the data, defining encoder mappings  $f^u$  and  $f^v$ . These map a single trial  $i$  of the data  $(\cdot, \text{in})^{(i)} \in \mathcal{X}$  to  $(\boldsymbol{\xi}_t^{(i)})_u$  and  $(\boldsymbol{\xi}_t^{(i)})_v$ .

We now perform a multinomial regression from  $(\boldsymbol{\xi}_t^{(i)})_u$  to  $(\boldsymbol{\xi}_t^{(i)})_v$ .

$$\mathbf{p}_t^{(i)} = h\left(\left(\boldsymbol{\xi}_t^{(i)}\right)_u\right) \quad (1)$$

$$h(\boldsymbol{\xi}) = \sigma(W\boldsymbol{\xi} + \mathbf{b}) \quad (2)$$

where  $W \in \mathbb{R}^{M(v) \times M(u)}$ ,  $\mathbf{b} \in \mathbb{R}^{M(v)}$  and  $\sigma$  is the softmax. During training we sample states from the target PMFs  $(z_t^{(i)})_v \sim (\boldsymbol{\xi}_t^{(i)})_v$  thus arriving at a more well know problem scenario: classification of  $M(v)$ -classes. We optimize  $W$  and  $\mathbf{b}$  to minimise a cross-entropy loss to the target  $(\hat{z}_t^{(i)})_v$  using the `fit()` method of `sklearn.linear_model.LogisticRegression`.

We define decoding error, as the average Kullback-Leibler divergence  $D_{KL}$  between target and predicted distributions:

$$\mathcal{D}_{u \rightarrow v} := \frac{1}{S^{\text{test}}T} \sum_{i \in \text{test}} \sum_{t=1}^T D_{KL}\left(\mathbf{p}_t^{(i)}, (\boldsymbol{\xi}_t^{(i)})_v\right) \quad (3)$$

where  $D_{KL}$  is implemented with `scipy.special.rel_entr`.

In S3 Fig. the data  $X$  is sampled from a single teacher HMM, T, and we evaluate  $\mathcal{D}_{T \rightarrow S}$  and  $\mathcal{D}_{S \rightarrow T}$  for each student notated simply as S.