Discrete Latent Variable Representations for Low-Resource Text Classification

Shuning Jin[1*] **Sam Wiseman**[2] **Karl Stratos**[1] **Karen Livescu**[2]
[1]Rutgers University [2]Toyota Technological Institute at Chicago
shuning.jin@rutgers.edu, {swiseman,klivescu}@ttic.edu, stratos@cs.rutgers.edu

Abstract

While much work on deep latent variable models of text uses continuous latent variables, discrete latent variables are interesting because they are more interpretable and typically more space efficient. We consider several approaches to learning discrete latent variable models for text in the case where exact marginalization over these variables is intractable. We compare the performance of the learned representations as features for low-resource document and sentence classification. Our best models outperform the previous best reported results with continuous representations in these low-resource settings, while learning significantly more compressed representations. Interestingly, we find that an amortized variant of Hard EM performs particularly well in the lowest-resource regimes.[1]

1 Introduction

Deep generative models with latent variables have become a major focus of NLP research over the past several years. These models have been used both for generating text (Bowman et al., 2016) and as a way of learning latent representations of text for downstream tasks (Yang et al., 2017; Gururangan et al., 2019). Most of this work has modeled the latent variables as being continuous, that is, as vectors in $\mathbb{R}^d$, in part due to the simplicity of performing inference over (certain) continuous latents using variational autoencoders and the reparameterization trick (Kingma and Welling, 2014; Rezende et al., 2014).

At the same time, deep generative models with *discrete* latent variables are attractive because the latents are arguably more interpretable, and because they lead to significantly more compressed representations: A representation consisting of M floating point values conventionally requires $M \times 32$ bits, whereas M integers in $\{1, \ldots, K\}$ requires only $M \times \log_2 K$ bits.

Unfortunately, discrete latent variable models have a reputation for being more difficult to learn. We conduct a thorough comparison of several popular methods for learning such models, all within the framework of maximizing the evidence lower bound (ELBO) on the training data. In particular, we compare learning such models either with a Vector Quantized-VAE (van den Oord et al., 2017, VQ-VAE), a more conventional VAE with discrete latent variables (Jang et al., 2017; Maddison et al., 2017), or with an amortized version of "Hard" or "Viterbi" Expectation Maximization (Brown et al., 1993), which to our knowledge has not been explored to date. We consider both models where the latents are local (i.e., per token) and where they are global (i.e., per sentence); we assess the quality of these learned discrete representations as features for a low-resource text classifier, as suggested by Gururangan et al. (2019), and in a nearest neighbor-based retrieval task.

Our classification experiments distinguish between (1) the setting where the classifier must consume only the discrete representation associated with each sentence (i.e., the discrete assignment that maximizes the approximate posterior), and (2) the setting where the classifier may consume the *embeddings* of this discrete representation learned by the VAE encoder. Note that the former setting is more flexible, since we need only store a sentence's discrete representation, and are therefore free to use task-specific (and possibly much smaller) architectures for classification. In case (1), we are able to effectively match the performance of Gururangan et al. (2019) and other baselines; in case (2), we outperform them. Our experiments also suggest that Hard EM performs particularly well in

[*] Work done as an intern at Toyota Technological Institute at Chicago.

[1] Code available on GitHub: https://github.com/shuningjin/discrete-text-rep

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4831–4842
July 5 - 10, 2020.

case (1) when there is little supervised data, and that VQ-VAE struggles in this setting.

2 Related Work

Our work builds on recent advances in discrete representation learning and its applications. In particular, we are inspired by recent success with VQ-VAEs outside NLP (van den Oord et al., 2017; Razavi et al., 2019). These works show that we can generate realistic speech and image samples from discrete encodings, which better align with symbolic representations that humans seem to work with (e.g., we naturally encode continuous speech signals into discrete words). Despite its success in speech and vision, VQ-VAE has not been considered as much in NLP. One exception is the translation model of Kaiser et al. (2018) that encodes a source sequence into discrete codes using vector quantization. But their work focuses on making inference faster, by decoding the target sequence from the discrete codes non-autoregressively. To our knowledge, we are the first that explores general text representations induced by VQ-VAEs for semi-supervised and transfer learning in NLP.

In addition to exploring the viability of VQ-VAEs for text representation learning, an important part of this paper is a systematic comparison between different discretization techniques. Gumbel-Softmax (Jang et al., 2017; Maddison et al., 2017) is a popular choice that has been considered for supervised text classification (Chen and Gimpel, 2018) and dialog generation (Zhao et al., 2018). In the binary latent variable setting, straight-through estimators are often used (Dong et al., 2019). Another choice is "continuous decoding" which takes a convex combination of latent values to make the loss differentiable (Al-Shedivat and Parikh, 2019). Yet a less considered choice is Hard EM (Brown et al., 1993; De Marcken, 1995; Spitkovsky et al., 2010). A main contribution of this work is a thorough empirical comparison between such different choices in a controlled setting.

To demonstrate the usefulness of our models, we focus on improving low-resource classification performance by pretraining on unlabeled text. Previous best results are obtained with continuous latent-variable VAEs, e.g., VAMPIRE (Gururangan et al., 2019). We show that our discrete representations outperform these previous results while being significantly more lightweight.

3 Background

We consider generative models of a sequence $x = x_{1:T}$ of T word tokens. We assume our latents to be a sequence $z = z_{1:L}$ of L discrete latent vectors, each taking a value in $\{1, \dots, K\}^M$; that is, $z \in \{1, \dots, K\}^{M \times L}$. As is common in VAE-style models of text, we model the text autoregressively, and allow arbitrary interdependence between the text and the latents. That is, we have $p(x, z; \boldsymbol{\theta}) = p(z) \times \prod_{t=1}^{T} p(x_t \mid x_{<t}, z; \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ are the generative model's parameters. We further assume $p(z)$ to be a fully factorized, uniform prior: $p(z) = \frac{1}{K^{ML}}$.

Maximizing the marginal likelihood of such a model will be intractable for moderate values of K, M, and L. So we consider learning approaches that maximize the ELBO (Jordan et al., 1999) in an amortized way (Kingma and Welling, 2014; Rezende et al., 2014):

$$\mathrm{ELBO}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbb{E}_{q(z \mid x; \boldsymbol{\phi})} \left[\log \frac{p(x, z; \boldsymbol{\theta})}{q(z \mid x; \boldsymbol{\phi})} \right],$$

where $q(z \mid x; \boldsymbol{\phi})$ is the approximate posterior given by an inference or encoder network with parameters $\boldsymbol{\phi}$. The approaches we consider differ in terms of how this approximate posterior q is defined.

Mean-Field Categorical VAE (CatVAE) A standard Categorical VAE parameterizes the approximate posterior as factorizing over categorical distributions that are independent given x. We therefore maximize:

$$\begin{aligned} &\mathbb{E}_{q(z \mid x; \boldsymbol{\phi})} [\log p(x \mid z; \boldsymbol{\theta})] - \sum_{m,l} \mathrm{KL}(q_{ml} || p_{ml}) \\ &= \mathbb{E}_{q(z \mid x; \boldsymbol{\phi}))} [\log p(x \mid z; \boldsymbol{\theta})] \\ &\quad + \sum_{m,l} H(q_{ml}) - ML \log K, \end{aligned}$$

where $q(z \mid x; \boldsymbol{\phi}) = \prod_{m=1}^{M} \prod_{l=1}^{L} q_{ml}(z_{ml} \mid x; \boldsymbol{\phi})$, $p_{ml} = 1/K$, and H is the entropy.

We approximate the expectation above by sampling from the q_{ml}, and we use the straight-through gradient estimator (Bengio et al., 2013; Jang et al., 2017) to compute gradients with respect to $\boldsymbol{\phi}$. We find this approach to be more stable than using the REINFORCE (Williams, 1992) gradient estimator, or a Concrete (Maddison et al., 2017; Jang et al., 2017) approximation to categorical distributions. Specifically, we sample from a categorical distribution using the Gumbel-Max trick (Maddison et al., 2014) in the forward pass, and approximate the gradient using softmax with a small temperature. This approach is also referred to as straight-through Gumbel-Softmax (Jang et al., 2017).

VQ-VAE A VQ-VAE (van den Oord et al., 2017; Razavi et al., 2019) can also be seen as maximizing the ELBO, except the approximate posterior is assumed to be a point mass given by

$$q_{ml}(z_{ml}|x) = \begin{cases} 1 & \text{if } z_{ml} = \hat{z}_{ml} \\ 0 & \text{otherwise} \end{cases},$$

where

$$\hat{z}_{ml} = \underset{j\in\{1,\dots,K\}}{\arg\min} ||\mathbf{e}_j^{(m)} - \text{enc}(x)_{ml}||_2, \qquad (1)$$

and $\mathbf{e}_j^{(m)} \in \mathbb{R}^d$ is an embedding of the j^{th} discrete value z_{ml} can take on, and $\text{enc}(x)_{ml} \in \mathbb{R}^d$ is an encoding corresponding to the ml^{th} latent given by an encoder network. These $\mathbf{e}_j^{(m)}$ embedding vectors are often referred to as a VQ-VAE's "code book". In our setting, a code book is shared across latent vectors.

VQ-VAEs are typically learned by maximizing the ELBO assuming degenerate approximate posteriors as above, plus two terms that encourage the encoder embeddings and the "code book" embeddings to become close. In particular, we attempt to maximize the objective:

$$\log p(x \mid \hat{z}) - \sum_{m,l} ||\text{sg}(\text{enc}(x)_{ml}) - \mathbf{e}^{(m)}_{\hat{z}_{m,l}}||_2^2 \qquad (2)$$
$$- \beta \sum_{m,l} ||\,\text{enc}(x)_{ml} - \text{sg}(\mathbf{e}^{(m)}_{\hat{z}_{m,l}})||_2^2,$$

where sg is the stop-gradient operator, and $\hat{z} = \hat{z}_{1:L}$ is the sequence of minimizing assignments $\hat{z}_{m,l}$ for each $\text{enc}(x)_{ml}$. The loss term following the β is known as the "commitment loss". Gradients of the likelihood term with respect to $\text{enc}(x)$ are again estimated with the straight-through gradient estimator.

Hard EM We train with an amortized form of Hard EM. First we define a relaxed version of z, $\tilde{z}$, where each $\tilde{z}_{ml}$ is a softmax over K outputs (rather than a hard assignment) and is produced by an inference network with parameters ϕ.[2] In the E-Step, we take a small, constant number of gradient steps to maximize $\log p(x \mid \tilde{z}; \boldsymbol{\theta})$ with respect to $\boldsymbol{\phi}$ (for a fixed $\boldsymbol{\theta}$). In the M-Step, we take a single gradient step to maximize $\log p(x \mid \hat{z}; \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$, where $\hat{z}$ contains the element-wise argmaxes of $\tilde{z}$ as produced by the inference network (with its most recent parameters $\boldsymbol{\phi}$). Thus, Hard EM can also be interpreted as maximizing the (relaxed) ELBO. We also note that taking multiple steps in the hard E-step somewhat resembles the recently proposed aggressive training of VAEs (He et al., 2019).

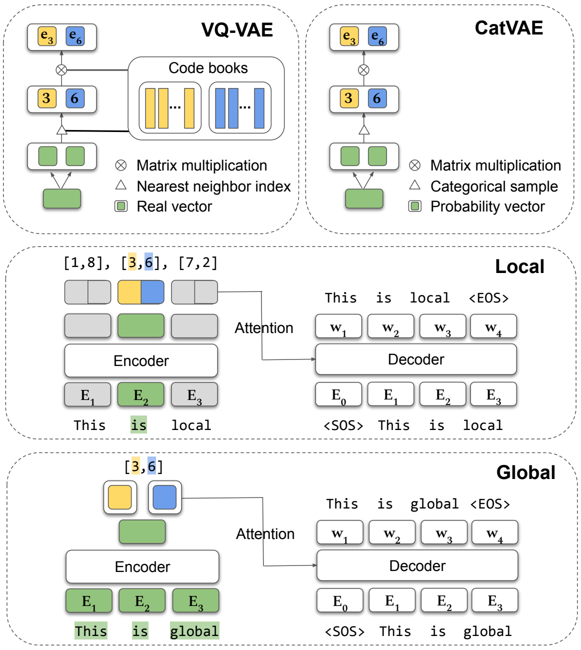

Figure 1: Discrete VAE architectures with $M = 2$. The **Local** (middle) and **Global** (bottom) models are two different encoder-decoder setups. The top row shows the procedure of converting continuous output from encoder into discrete input to decoder by drawing discrete samples: **VQ-VAE** (top left) draws samples from point mass distributions using nearest neighbor lookup from the code books; **CatVAE** (top right) samples from categorical distributions directly.

4 Models and Architectures

Recall that the latent sequence is $z = z_{1:L}$, where $z_l \in \{1, \dots, K\}^M$. We consider two generative models $p(x \mid z; \boldsymbol{\theta})$, one where $L = T$ and one where $L = 1$. Each latent in the former model corresponds to a word, and so we refer to this as a "local" model, whereas in the second model we view the latents as being "global", since there is one latent vector for the whole sentence. We use the following architectures for our encoders and decoder, as illustrated in Figure 1.

[2] Note this assumes our generative model can condition on such a relaxed latent variable.

4.1 Encoder

The encoder (parameterized by ϕ) maps an example x to the parameters of an approximate posterior distribution. Our encoder uses a single-layer Transformer (Vaswani et al., 2017) network to map $x = x_{1:T}$ to a sequence of T vectors $\mathbf{h}_1, \ldots, \mathbf{h}_T$, each in $\mathbb{R}^d$.

Mean-Field Categorical VAE For the local model, we obtain the parameters of each categorical approximate posterior q_{mt} as $\mathrm{softmax}(\mathbf{W}_m\, \mathbf{h}_t)$, where each $\mathbf{W}_m \in \mathbb{R}^{K \times d}$ is a learned projection. For the global model, we obtain the parameters of each categorical approximate posterior q_{m1} as $\mathrm{softmax}\left(\frac{\sum_t \mathbf{W}_m\, \mathbf{h}_t}{T}\right)$; that is, we pass token-level $\mathbf{h}_t$ vectors through learned projections $\mathbf{W}_m$, followed by mean-pooling.

VQ-VAE For the local model, let $\tilde{d} = d/M$. We obtain $\mathrm{enc}(x)_{mt}$, the encoding of the mt^{th} latent variable, as $\mathbf{h}_{t,(m-1)\tilde{d}:m\tilde{d}}$, following Kaiser et al. (2018). That is, we take the m^{th} $\tilde{d}$-length subvector of $\mathbf{h}_t$. For the global model, let $\tilde{d} = d$. We first project $\mathbf{h}_t$ to $\mathbb{R}^{Md}$, mean-pool, and obtain $\mathrm{enc}(x)_{m1}$ by taking the m^{th} $\tilde{d}$-length subvector of the resulting pooled vector. A VQ-VAE also requires learning a code book, and we define M code books $\mathbf{E}^{(m)} = [\mathbf{e}_1^{(m)\top}; \ldots; \mathbf{e}_K^{(m)\top}] \in \mathbb{R}^{K \times \tilde{d}}$.

Hard EM We use the same encoder architecture as in the mean-field Categorical VAE case. Note, however, that we do not sample from the resulting categorical distributions. Rather, the $\mathrm{softmax}$ distributions are passed directly into the decoder.

4.2 Decoder

In the case of the mean-field Categorical VAE, we obtain a length-L sequence of vectors $z_l \in \{1, \ldots, K\}^M$ after sampling from the approximate posteriors. For the VQ-VAE, on the other hand, we obtain the sequence of $\hat{z}_l$ vectors by taking the indices of the closest code book embeddings, as in Equation (1).

In both cases, the resulting sequence of discrete vectors is embedded and consumed by the decoder. In particular, when learning with a VQ-VAE, the embedding of $\hat{z}_{ml}$ is simply $\mathbf{e}^{(m)}_{\hat{z}_{ml}}$, whereas for the Categorical VAE each discrete latent is embedded using a trained embedding layer. In the local model, when $M > 1$, we concatenate the M embeddings to form a single real vector embedding for the l^{th} latent variable. In the global model, we use the M embeddings directly. This resulting sequence of T or M real vectors is then viewed as the source side input for a standard 1-layer Transformer encoder-decoder model (Vaswani et al., 2017), which decodes x using causal masking.

As above, for Hard EM, we do not obtain a sequence of discrete vectors from the encoder, but rather a sequence of $\mathrm{softmax}$ distributions. These are multiplied into an embedding layer, as in the Categorical VAE case, and fed into the Transformer encoder-decoder model.

5 Evaluating Latent Representations

Similar to Gururangan et al. (2019), we evaluate the learned latent representations by using them as features in a text classification system. We are in particular interested in using latent representations learned on unlabeled text to help improve the performance of classifiers trained on a small amount of labeled text. Concretely, we compare different discrete latent variable models in following steps:

1. Pretraining an encoder-decoder model on in-domain unlabeled text with an ELBO objective, with early stopping based on validation perplexity.

2. Fixing the encoder to get discrete latents for the downstream classification task, and training a small number of task-specific parameters on top, using varying amounts of labeled data. As noted in the introduction, we consider both reembedding these latents from scratch, or using the embeddings learned by the encoder.

5.1 Tasks and Datasets

The datasets we use for classification are AG News, DBPedia, and Yelp Review Full (Zhang et al., 2015), which correspond to predicting news labels, Wikipedia ontology labels, and the number of Yelp stars, respectively. The data details are summarized in Table 1. For all datasets, we randomly sample 5,000 examples as development data. To evaluate the efficiency of the latent representation in low-resource settings, we train the classifier with varying numbers of labeled instances: 200, 500, 2500, and the full training set size (varies by dataset). We use accuracy as the evaluation metric.

In preprocessing, we space tokenize, lowercase, and clean the text as in Kim (2014), and then truncate each sentence to a maximum sequence length of 400. For each dataset, we use a vocabulary of the 30,000 most common words.

Dataset	# Classes	Train	Dev	Test
AG News	4	115K	5K	7.6K
DBPedia	14	555K	5K	70K
Yelp Review Full	5	645K	5K	50K

Table 1: The number of classes and the numbers of examples in each data subset, for the classification tasks.

5.2 Transfer Paradigm

When transferring to a downstream classification task, we freeze the pretrained encoder and add a lightweight classifier on top, viewing each sentence as an L-length sequence of vectors in $\{1, \dots, K\}^M$, as described in Section 4. For instance, the sentence (from the DBPedia dataset) *"backlash is a 1986 australian film directed by bill bennett"* is encoded as [90, 114, 30, 111] under a global model with $M = 4$, and as [[251, 38], [44, 123], [94, 58], [228, 53], [88, 55], [243, 43], [66, 236], [94, 72], [172, 61], [236, 150]] under a local model with $M = 2$.

As noted in the introduction, we consider two ways of embedding the integers for consumption by a classifier. We either (1) learn a new task-specific embedding space $\mathbf{E}^{(m)}_{\text{task}}$ (i.e., reembedding) or (2) use the fixed embedding space $\mathbf{E}^{(m)}$ from pretraining. The first setting allows us to effectively replace sentences with their lower dimensional discrete representations, and learn a classifier on the discrete representations from scratch. In the local model, we obtain token-level embedding vectors by concatenating the M subvectors corresponding to each word. The resulting embeddings are either averaged, or fed to a Transformer and then averaged, and finally fed into a linear layer followed by a $\mathrm{softmax}$.

6 Experimental Details

6.1 Baselines

We first experiment with three common text models: CBOW (Mikolov et al., 2013), bidirectional LSTM (Hochreiter and Schmidhuber, 1997), and a single-layer Transformer encoder. We find CBOW (with 64-dimensional embeddings) to be the most robust in settings with small numbers of labeled instances, and thus report results only with this baseline among the three. Further, we compare to VAMPIRE (Gururangan et al., 2019), a framework of pretraining VAEs for text classification using continuous latent variables. We pretrain VAMPIRE models on in-domain text for each dataset with 60 random hyperparameter search (with same ranges as specified in their Appendix A.1), and select best models based on validation accuracy in each setting.

6.2 Hyperparameters

In our experiments, we use Transformer layers with $d_{\text{model}} = 64$. For optimization, we use Adam (Kingma and Ba, 2015), either with a learning rate of 0.001 or with the inverse square-root schedule defined in Vaswani et al. (2017) in pretraining. We use a learning rate of 0.0003 in classification. We tune other hyperparameters with random search and select the best settings based on validation accuracy. For the latent space size, we choose M in $\{1, 2, 4, 8, 16\}$ and K in $\{128, 256, 512, 1024, 4096\}$. Model specific hyperparameters are introduced below.

6.3 VQ-VAE

In VQ-VAE, an alternative to the objective in Equation (2) is to remove its second term, while using an auxiliary dictionary learning algorithm with exponential moving averages (EMA) to update the embedding vectors (van den Oord et al., 2017). We tune whether to use EMA updates or not. Also, we find small β for commitment loss to be beneficial, and search over $\{0.001, 0.01, 0.1\}$.

6.4 Mean-Field Categorical VAE

We find that using the discrete analytic KL divergence term directly in the ELBO objective leads to posterior collapse. The KL term vanishes to 0 and the q_{ml} distributions converge to the uniform priors. To circumvent this, we modify the KL term to be $\max(\text{KL}, \lambda)$. This is known as Free Bits (Kingma et al., 2016; Li et al., 2019), which ensures that the latent variables encode a certain amount of information by not penalizing the KL divergence when it is less than λ. We set $\lambda = \gamma ML \log K$, where γ is a hyperparameter between 0 and 1. That is, we allocate a "KL budget" as a fraction of $ML \log K$, which is the upper bound of KL divergence between ML independent categorical distributions and uniform prior distributions. Since in this case $\text{KL}(q_{ml}(z_{ml} \mid x) || p_{ml}(z_{ml})) = \log K - H[q_{ml}(z_{ml} \mid x)]$, this is equivalent to thresholding $H[q_{ml}(z_{ml} \mid x)]$ by $(1 - \gamma) \log K$. We experiment with $\gamma \in \{0.2, 0.4, 0.6, 0.8, 1\}$.[3]

[3] Note that when $\gamma \geq 1$ the VAE reduces to an autoencoder.

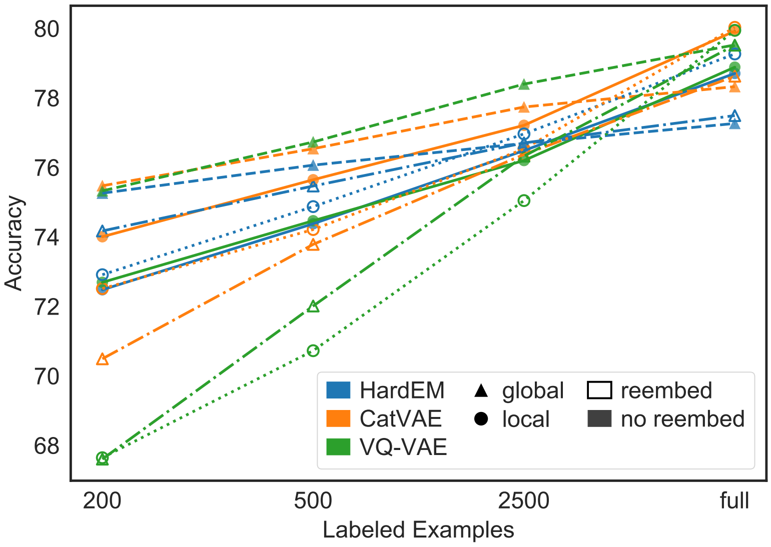

Figure 2: The accuracies obtained by Hard EM, Categorical VAE, and VQ-VAE representations, averaged over the AG News, DBPedia, and Yelp Full development datasets, for different numbers of labeled training examples. Triangular and circular markers correspond to global and local models, respectively. Unshaded and shaded markers correspond to reembedding from scratch and using encoder embeddings, respectively.

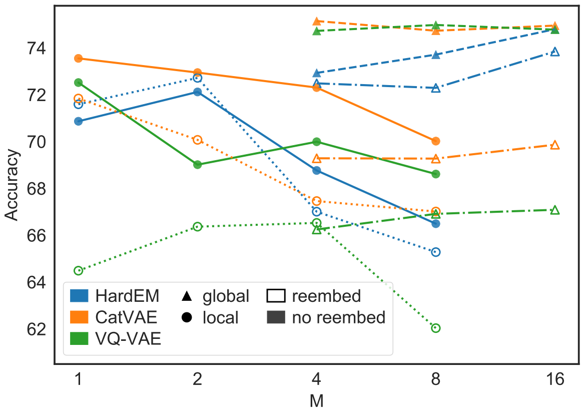

Figure 3: The averaged accuracies obtained from using Hard EM, Categorical VAE, and VQ-VAE representations and 200 labeled examples, for different M values.

6.5 Hard EM

We vary the number of gradient steps in the E-step in $\{1, 3\}$. At evaluation time, we always take the argmax of $\tilde{z}$ to get a hard assignment.

7 Results

In Figure 2, we compare the accuracy obtained by the representations from our Hard EM, Categorical VAE, and VQ-VAE models, averaged over the development datasets of AG News, DBPedia, and Yelp Full. In particular, we plot the best accuracy obtained over all hyperparameters (including M) for different numbers of labeled examples; we distinguish between local and global models, and between when the discrete representations are reembedded from scratch and when the encoder embeddings are used.

We see that using the encoder embeddings typically outperforms reembedding from scratch, and that global representations tend to outperform local ones, except in the full data regime. Furthermore, we see that the Categorical VAE and VQ-VAE are largely comparable on average, though we undertake a finer-grained comparison by dataset in Appendix A. Perhaps most interestingly, we note that when reembedding from scratch, Hard EM significantly outperforms the other approaches in the lowest data regimes (i.e., for 200 and 500 examples). In fact, Hard EM allows us to match the performance of the best previously reported results even when reembedding from scratch; see Table 3.

Table 2 shows the best combinations of model and hyperparameters when training with 200 labeled examples on AG News. These settings were used in obtaining the numbers in Figure 2, and are largely stable across datasets.

In Figure 3, we compare the average accuracy of our local and global model variants trained on 200 labeled examples, as we vary M. When reembedding, local representations tend to improve as we move from $M = 1$ to $M = 2$, but not significantly after that. When reembedding global representations, performance increases as M does. Unsurprisingly, when not reembedding, M matters less.

	Method	K	M
Local	CatVAE	4096	1
Local (re)	Hard EM	1024	1
Global	CatVAE	256	4
Global (re)	Hard EM	4096	4

Table 2: Best methods and settings of K and M when training on 200 labeled examples of the AG News corpus and evaluating on the development set. The "(re)" affix indicates that latent variables are reembedded from scratch.

Model	200	500	2500	Full
AG News				
CBOW	63.4 (1.5)	72.9 (0.7)	82.1 (0.2)	90.0 (0.2)
VAMPIRE*	83.9 (0.6)	84.5 (0.4)	85.8 (0.2)	-
VAMPIRE	82.2 (0.8)	84.7 (0.2)	**86.4** (0.4)	**91.0** (0.1)
Local	82.7 (0.1)	84.3 (0.3)	85.0 (0.4)	86.6 (0.2)
Local (re)	82.7 (0.4)	84.0 (0.3)	85.4 (0.1)	87.1 (0.3)
Global	**84.6** (0.1)	**85.7** (0.1)	86.3 (0.2)	87.5 (0.6)
Global (re)	83.9 (0.5)	84.6 (0.2)	85.1 (0.3)	86.9 (0.1)
DBPedia				
CBOW	72.7 (0.6)	84.7 (0.7)	92.8 (0.3)	97.7 (0.1)
VAMPIRE	89.1 (1.3)	93.7 (0.5)	**95.7** (0.2)	**98.2** (0.1)
Local	89.2 (0.2)	92.8 (0.4)	94.6 (0.2)	97.1 (0.3)
Local (re)	88.7 (0.2)	90.2 (0.3)	93.3 (0.1)	96.9 (0.2)
Global	**91.8** (0.5)	**94.3** (0.3)	95.0 (0.2)	95.6 (0.0)
Global (re)	88.5 (0.7)	92.3 (0.7)	94.6 (0.4)	95.8 (0.1)
Yelp Full				
CBOW	31.0 (5.9)	41.1 (0.6)	48.4 (0.4)	58.9 (0.4)
VAMPIRE	41.4 (2.9)	47.2 (0.7)	52.5 (0.1)	**60.3** (0.1)
Local	46.2 (0.8)	49.0 (0.5)	51.9 (0.5)	53.1 (0.3)
Local (re)	47.2 (0.7)	49.4 (0.7)	52.1 (0.2)	55.0 (0.6)
Global	**48.5** (1.0)	**50.1** (0.5)	**53.0** (0.3)	54.9 (0.4)
Global (re)	46.0 (0.5)	47.4 (0.5)	48.8 (0.8)	53.8 (0.3)

Table 3: Test accuracy results by dataset and by the number of labeled examples used in training. The scores are averages over five random subsamples, with standard deviations in parentheses and column bests in **bold**. VAMPIRE* for AG News is reported by Gururangan et al. (2019) and VAMPIREs are from our experiments.

Finally, we show the final accuracies obtained by our best models on the test data of each dataset in Table 3. We see that on all datasets when there are only 200 or 500 labeled examples, our best model outperforms VAMPIRE and the CBOW baseline, and our models that reembed the latents from scratch match or outperform VAMPIRE. As noted in Table 2, it is Hard EM that is particularly performant in these settings.

8 Analysis and Discussion

8.1 Qualitative analysis

To gain a better understanding of what the learned clusters represent, we examine their patterns on the AG News dataset labeled with four classes. Since VQ-VAEs and Categorical VAEs exhibit similar patterns, we focus on the latter model.

Tables 4 and 5 show examples of sentence- and word-level clusters, respectively, induced by Categorical VAEs. The sentence-level model encodes each document into $M = 4$ latents, each taking one of $K = 256$ integers. The word-level model encodes each word into $M = 1$ latent taking one of $K = 1024$ integers. Since a word can be assigned multiple clusters, we take the majority cluster for illustration purposes.

We see that clusters correspond to topical aspects of the input (either a document or a word). In particular, in the sentence-level case, documents in the same cluster often have the same ground-truth label. We also find that each of M latents independently corresponds to topical aspects (e.g., $z_1 = 65$ implies that the topic has to do with technology); thus, taking the combination of these latents seems to make the cluster "purer". The word-level clusters are also organized by topical aspects (e.g., many words in cluster 510 are about modern conflicts in the Middle East).

8.2 Effect of Alternating Optimization

While Hard EM achieves impressive performance when reembedding from scratch and when training on only 200 or 500 examples, we wonder whether this performance is due to the alternating optimization, to the multiple E-step updates per M-step update, or to the lack of sampling. We accordingly experiment with optimizing our VQ-VAE and CatVAE variants in an alternating way, allowing multiple inference network updates per update of the generative parameters $\boldsymbol{\theta}$. We show the results on the AG News dataset in Table 6. We find that alternating does generally improve the performance of VQ-VAE and CatVAE as well, though Hard EM performs the best overall when reembedding from scratch. Furthermore, because Hard EM requires no sampling, it is a compelling alternative to CatVAE. For all three methods, we find that doing 3 inference network update steps during alternating optimization performs no better than doing a single one, which suggests that aggressively optimizing the inference network is not crucial in our setting.

Cluster	Class	Text
(23, 155, 24, 53)	World	a platoon in iraq is being investigated for allegedly refusing to carry out a convoy mission...
	World	afp chechen warlord shamil basayev has claimed responsibility for the deadly school...
	World	the federal government has sent a team of defence personnel to verify a claim that two...
	World	an audio tape purportedly by osama bin laden praises gunmen who attacked a us consulate...
(41, 75, 175, 222)	Business	amazon com says it has reached an agreement to buy joyo com, the largest internet retailer...
	Business	electronic data systems offered voluntary early retirement to about 9, 200 us employees...
	Business	in the aftermath of its purchase of at amp t wireless, cingular wireless is selling several sets...
	Sci/Tech	wired amp wireless continues its reign at the top spot among it priorities due to widespread...
(10, 208, 179, 180)	Sports	this is the week of the season when every giants defensive back needs to have shoulders as...
	Sports	drew henson will have to wait before he's the star of the dallas cowboys offense right now...
	Sports	st louis how do you beat the greatest show on turf with two rookie cornerbacks...
	Sports	cincinnati bengals coach marvin lewis said yesterday that he expects quarterback carson...
(65, 224, 78, 114)	Sci/Tech	microsoft acknowledged on monday it continued to battle a technical glitch that prevented...
	Sci/Tech	users of the music player should watch out for hacked themes a flaw allows would be...
	World	microsoft's popular internet explorer has a serious rival in the firefox browser
	Sci/Tech	microsoft has doubled the period of time it will allow business users of windows xp to...

Table 4: Examples of sentence-level ($M = 4$, $K = 256$) clusters on AG News.

Cluster	Words
822	government indonesia guilty prison general prosecutors leader law german sex authorities charged marched issue
651	yankees veteran baltimore quarterback offense tampa steelers giants defensive cleveland minnesota pittsburgh
595	month currency low session dollar euro greenback yen monetary weakening lows versus maintained grip rebounded
305	if despite when although
304	core plans intel athlon opteron processors chip hewlett packard strategy clearer forum designs desktop upped ante
802	bit cameras image pleasing integrates multimedia functions gprs automation self types btx supercomputers logic
298	president dick cheney john republicans kerry voters democrat javier sen kellogg
994	exploded bomb near killing injuring explosion eight residents firefighters leak central philippine 55 heavily cancun
484	apple atari san francisco sony toshiba anaheim finally assault famed mp3 freedom u2 accusations brook introduces
510	iraq killed car rebel iraqi military suicide forces marines insurgents baghdad evacuation bomber strikes explosions

Table 5: Word-level ($M = 1$, $K = 1024$) clusters on AG News. We take the majority cluster for each word for illustration purposes.

Model	200	200 (re)	500	500 (re)
EM-Local	81.4	82.1	83.0	82.8
EM-Global	85.6	**84.6**	85.5	**85.4**
Cat-Local-Alt	83.3	82.9	84.8	84.1
Cat-Global-Alt	**86.4**	83.1	**87.1**	85.0
Cat-Local	83.2	82.5	85.3	84.8
Cat-Global	85.4	82.8	86.1	84.5
VQ-Local-Alt	82.9	81.1	84.8	81.4
VQ-Global-Alt	84.7	79.6	85.9	82.9
VQ-Local	82.6	78.7	83.6	81.3
VQ-Global	83.0	76.8	85.4	82.0

Table 6: Effect of alternating optimization on AG News classification with 200 and 500 labels. The "(re)" affix denotes reembedding. Accuracies are on development set with column highs in **bold**.

8.3 Compression

We briefly discuss in what sense discrete latent representations reduce storage requirements. Given a vocabulary of size 30,000, storing a T-length sentence requires $T \log_2 30000 \approx 14.9T$ bits. Our models require at most $ML \log_2 K$ bits to represent a sentence, which is generally smaller, and especially so when using a global representation. It is also worth noting that storing a d-dimensional floating point representation of a sentence (as continuous latent variable approaches might) costs $32d$ bits, which is typically much larger.

While the above holds for storage, the space required to *classify* a sentence represented as ML integers using a parametric classifier may not be smaller than that required for classifying a sentence represented as a d-dimensional floating point vector. On the other hand, nearest neighbor-based methods, which are experiencing renewed interest (Guu et al., 2018; Chen et al., 2019; Wiseman and Stratos, 2019), should be significantly less expensive in terms of time and memory when sentences are encoded as ML integers rather than d-dimensional floating point vectors. In the next subsection we quantitatively evaluate our discrete representations in a nearest neighbor-based retrieval setting.

	Discrete Embedding		
	M=4, K=256	M=8, K=128	M=16, K=256
Hard EM	76.1	79.6	78.8
CatVAE	77.5	73.7	78.5
VQ-VAE	69.1	73.5	71.2
	Continuous Embedding (300d)		
	$\mathcal{L}_2$	COSINE	
GloVe	76.4	76.6	
fastText	72.8	74.1	

Table 7: Unsupervised document retrieval on AG News dataset, measured by average label precision of top 100 nearest neighbors of the development set. Underlined score is the row best. Discrete representations use Hamming distance.

8.4 Nearest Neighbor-Based Retrieval

In the classification experiments of Section 5, we evaluated our discrete representations by training a small classifier on top of them. Here we evaluate our global discrete representations in a document retrieval task to directly assess their quality; we note that this evaluation does not rely on the learned code books, embeddings, or a classifier.

In these experiments we use each document in the development set of the AG News corpus as a query to retrieve 100 nearest neighbors in the training corpus, as measured by Hamming distance. We use average label precision, the fraction of retrieved documents that have the same label as the query document, to evaluate the retrieved neighbors. We compare with baselines that use averaged 300d pretrained word vectors (corresponding to each token in the document) as a representation, where neighbors are retrieved based on cosine or $\mathcal{L}_2$ distance. We use GloVe with a 2.2 million vocabulary (Pennington et al., 2014) and fastText with a 2 million vocabulary (Mikolov et al., 2018). The results are in Table 7. We see that CatVAE and Hard EM outperform these CBOW baselines (while being significantly more space efficient), while VQ-VAE does not. These results are in line with those of Figure 2, where VQ-VAE struggles when its code book vectors cannot be used (i.e., when reembedding from scratch).

In Figure 4 we additionally experiment with a slightly different setting: Rather than retrieving a fixed number of nearest neighbors for a query document, we retrieve all the documents within a neighborhood of Hamming distance $\leq D$, and calculate the average label precision. These results use global representations with $M = 16$, and we therefore examine thresholds of $D \in \{0, \ldots, 16\}$. We see that for CatVAE and Hard EM, the document similarity (or label precision) has an approximately linear correlation with Hamming distance. On the other hand, VQ-VAE shows a more surprising pattern, where high precision is not achieved until $D = 10$, perhaps suggesting that a large portion of the latent dimensions are redundant.

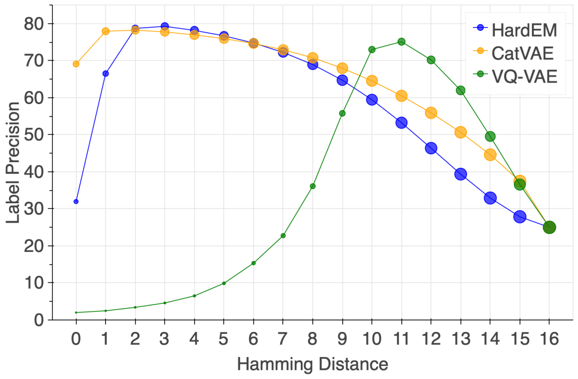

Figure 4: Retrieving document clusters with Hamming distance $\leq D$, for global models with $M = 16$ and $K = 256$. Query and target documents are from AG News's development set and training set respectively. Dot size indicates the number of documents in a cluster.

9 Conclusion

We have presented experiments comparing the discrete representations learned by a Categorical VAE, a VQ-VAE, and Hard EM in terms of their ability to improve a low-resource text classification system, and to allow for nearest neighbor-based document retrieval. Our best classification models are able to outperform previous work, and this remains so even when we reembed discrete latents from scratch in the learned classifier. We find that amortized Hard EM is particularly effective in low-resource regimes when reembedding from scratch, and that VQ-VAE struggles in these settings.

Acknowledgments

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-18-1-0166.

References

Maruan Al-Shedivat and Ankur Parikh. 2019. Consistency by agreement in zero-shot neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1184–1197, Minneapolis, Minnesota. Association for Computational Linguistics.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Mingda Chen and Kevin Gimpel. 2018. Smaller text classifiers with discriminative cluster embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 739–745, New Orleans, Louisiana. Association for Computational Linguistics.

Mingda Chen, Qingming Tang, Sam Wiseman, and Kevin Gimpel. 2019. A multi-task approach for disentangling syntax and semantics in sentence representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2453–2464.

Carl De Marcken. 1995. Lexical heads, phrase structure and the induction of grammar. In *Third Workshop on Very Large Corpora*.

Wei Dong, Qinliang Su, Dinghan Shen, and Changyou Chen. 2019. Document hashing with mixture-prior generative models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5229–5238.

Suchin Gururangan, Tam Dang, Dallas Card, and Noah A. Smith. 2019. Variational pretraining for semi-supervised text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5880–5894, Florence, Italy. Association for Computational Linguistics.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *Transactions of the Association for Computational Linguistics*, 6:437–450.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. 2019. Lagging inference networks and posterior collapse in variational autoencoders. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9:1735–1780.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with Gumbel-Softmax. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. 1999. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. 2018. Fast decoding in sequence models using discrete latent variables. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2390–2399, Stockholmsmssan, Stockholm Sweden. PMLR.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. 2016. Improving variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. 2019. A surprisingly effective fix for deep latent variable modeling of text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3601–3612.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The Concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of International Conference on Learning Representations (ICLR)*.

Chris J. Maddison, Daniel Tarlow, and Tom Minka. 2014. A* sampling. In *Advances in Neural Information Processing Systems*, pages 3086–3094.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pages 6306–6315.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Ali Razavi, Aaron van den Oord, and Oriol Vinyals. 2019. Generating diverse high-fidelity images with VQ-VAE-2. In *Advances in Neural Information Processing Systems*, pages 14866–14876.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1278–1286, Bejing, China. PMLR.

Valentin I. Spitkovsky, Hiyan Alshawi, Daniel Jurafsky, and Christopher D. Manning. 2010. Viterbi training improves unsupervised dependency parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 9–17, Uppsala, Sweden. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5363–5369.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3881–3890, Sydney, Australia. PMLR.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, pages 649–657.

Tiancheng Zhao, Kyusong Lee, and Maxine Eskenazi. 2018. Unsupervised discrete sentence representation learning for interpretable neural dialog generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1098–1107, Melbourne, Australia. Association for Computational Linguistics.

A Model Comparison by Datasets

We plot the development set classification performance of each method, this time distinguishing between datasets, in Figure 5.

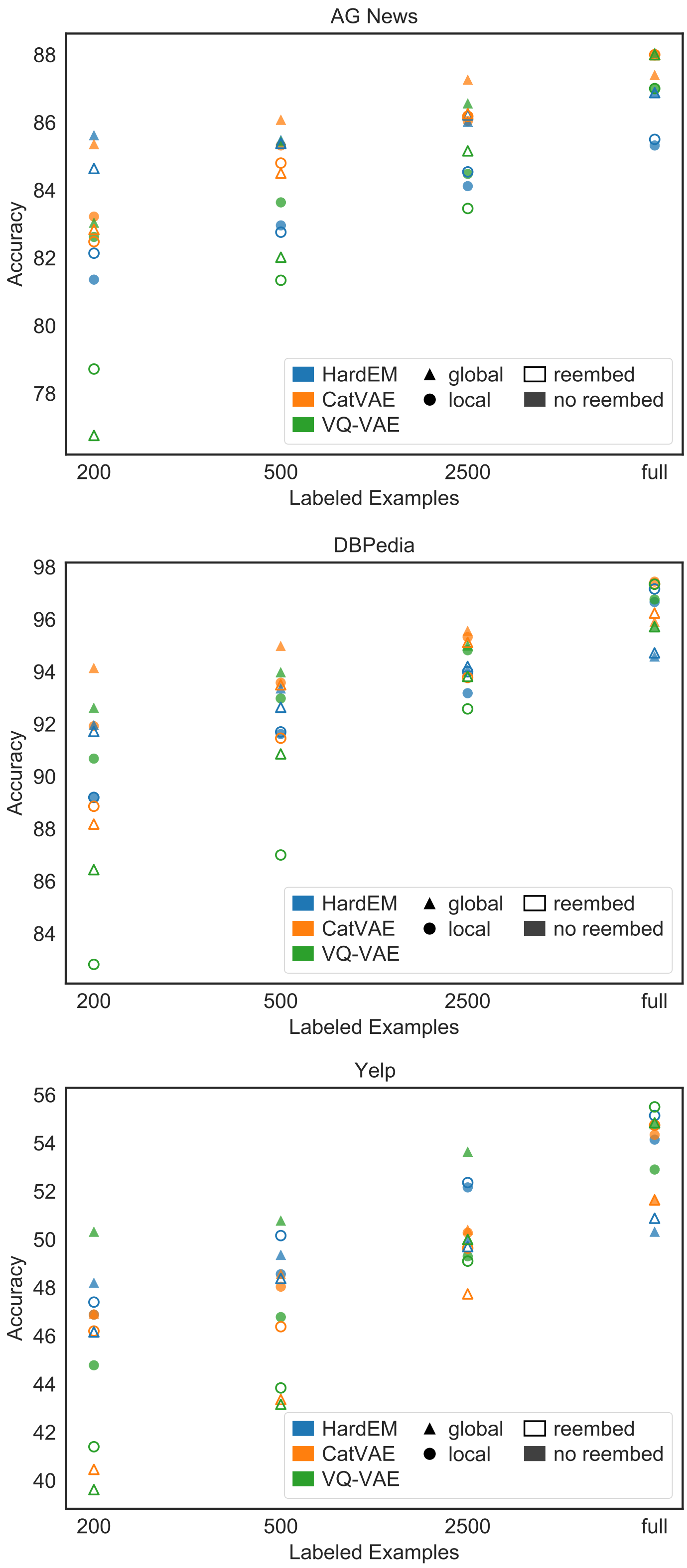

Figure 5: The accuracies obtained by Hard EM, Categorical VAE, and VQ-VAE representations on the development datasets of AG News (top), DBPedia (middle), and Yelp Full (bottom), for different numbers of labeled training examples. Triangular and circular markers correspond to global and local models, respectively. Unshaded and shaded markers correspond to reembedding from scratch and using encoder embeddings, respectively.