DS605 Assignment

Ananya (12040180), Vaibhav (12041650)

November 13, 2023

Contents

1 Problem Statement

The problem is to build an unsupervised machine translation system using discrete and continuous variance autoencoders. The key aspects are:

- It is an unsupervised setting, so there is no parallel data available, only monolingual corpora in the source and target languages.
- Variational autoencoders (VAEs) need to be used, which consist of an encoder network q(z—x) that encodes an input x into a latent representation z, and a decoder network p(x—z) that reconstructs x from z.
- Both discrete (categorical) and continuous (e.g. Gaussian) latent variables need to be explored. Discrete latent variables allow modeling discrete properties of language but are non-differentiable. Continuous variables are differentiable but may not capture discreteness well.
- The system needs to be able to translate from the source language to the target language in an unsupervised way, learning only from non-parallel monolingual corpora.
- Several sub-issues need to be addressed regarding the encoder, decoder, loss function, and training methodology when building this system.

So in summary, the key challenge is to build an unsupervised neural machine translation system using autoencoder models with both discrete and continuous latent variables, which presents challenges around modeling discrete sequences and non-differentiable training.

2 Problem Statement Formulation

The problem can be formulated as follows:

$$\text{Given: } \begin{cases} X = \{x^{(1)}, ..., x^{(m)}\}\text{: source language monolingual corpus} \\ Y = \{y^{(1)}, ..., y^{(n)}\}\text{: target language monolingual corpus} \end{cases} \tag{1}$$

$$\text{Learn: } \begin{cases} p_\theta(\bar{y}|x)\text{: source to target translation model} \\ p_\phi(\bar{x}|y)\text{: target to source translation model} \end{cases} \tag{2}$$

Where $\bar{x}$ and $\bar{y}$ are latent representations. This can be achieved using a VAE framework with encoder $q_\psi(\bar{z}|x)$ and decoder $p_\theta(x|\bar{z})$ networks.

The evidence lower bound (ELBO) objective is:

$$\mathcal{L}(\theta, \phi, \psi; X, Y) = \sum_{i=1}^{m} E_{q_\psi(\bar{y}|x^{(i)})}[\log p_\theta(x^{(i)}|\bar{y})] \tag{3}$$

$$-\text{KL}(q_\psi(\bar{y}|x^{(i)})||p(\bar{y})) \tag{4}$$

$$+\sum_{j=1}^{n} E_{q_\phi(\bar{x}|y^{(j)})}[\log p_\phi(y^{(j)}|\bar{x})] \tag{5}$$

$$-\text{KL}(q_\phi(\bar{x}|y^{(j)})||p(\bar{x})) \tag{6}$$

The key challenges are using discrete latent variables $\bar{z} \in \{1, ..., K\}$ and training with non-differentiable objectives.

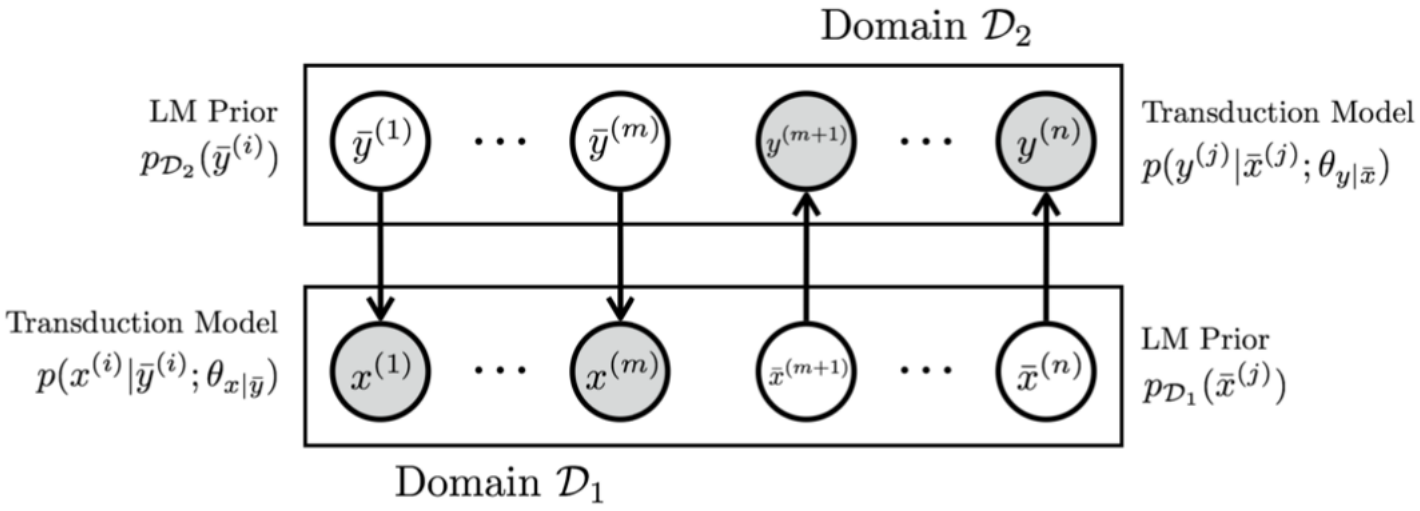

Figure 1: Proposed graphical model for style transfer via bitext completion. Shaded circles denote the observed variables and unshaded circles denote the latents. The generator is parameterized as an encoder-decoder architecture and the prior on the latent variable is a pretrained language model.

3 Building Unsupervised Machine Translation with VAEs

1. Preprocessing: Preprocess the monolingual corpora in the source and target languages to tokenize, clean, and normalize the data.

2. Language Model Training: Train source and target language models on the monolingual data to serve as priors in the VAE framework.

3. VAE Model Architecture:

1. For the encoder, utilize a bidirectional RNN (e.g., BiLSTM or GRU) to capture context from both directions.
2. For the decoder, use an attentional RNN decoder, similar to standard Seq2Seq models.
3. Loss function (Evidence Lower Bound, ELBO):

$$\log p(x) \geq E_{q(z|x)}[\log p(x|z)] - \mathrm{KL}(q(z|x)||p(z))$$

where KL term involves KL divergence between the approximate posterior $q(z|x)$ and the prior $p(z)$.

4. Training with Discrete Variables:

- Use approaches such as:
  - Gumbel-Softmax trick for a differentiable approximation to sampling from a categorical distribution.
  - REINFORCE algorithm with Monte Carlo sampling and rewards.
  - Heuristic approaches like stop-gradient to avoid propagating gradients through discrete variables.
- Additional considerations:
  - Implement parameter sharing between the encoder and decoder.
  - Ensure good initialization, such as pretraining the autoencoder, to avoid bad local minima.
  - Apply techniques like KL annealing, word dropout, and limiting latent space capacity to prevent posterior collapse and overfitting.

3.1 Variational Autoencoder (VAE)

VAEs learn deep generative models defined by a prior $p(z)$ and a conditional distribution $p_\theta(x|z)$. The marginal likelihood $p(x)$ is intractable, so VAEs optimize the evidence lower bound (ELBO):

$$L(x;\theta,\phi) = E_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - \beta\mathrm{KL}(q_\phi(z|x)||p(z))$$

where $q_\phi(z|x)$ is the approximate posterior parameterized by an inference network ϕ, and $p_\theta(x|z)$ is the generator network parameterized by θ. The KL divergence term is scaled by β for better training stability [2].

3.2 Posterior Collapse

Posterior collapse occurs when the KL term goes to 0, indicating the model ignores the latent code. Mitigation techniques include:

- KL annealing: Slowly increase β from 0 to 1 [3].
- Word dropout: Randomly drop words from the source [3].
- Free bits: Constrain KL to be above a threshold [4].

3.3 Architecture

We use an attentional encoder-decoder with GRUs, similar to [**?**]. The encoder maps an input x to a continuous latent code z via the approximate posterior $q_\phi(z|x)$. The decoder models $p_\theta(x|z)$ to reconstruct x from z. The model assumes each observed sentence $x^{(i)}$ or $y^{(j)}$ is generated from a latent sentence in the opposite domain, as in Figure 1 of [1].

The transduction distributions $p(x|z)$ and $p(y|z)$ share parameters θ. The priors $p(z)$ are pretrained language models, facilitating learning coupling between domains.

4 Answer to sub-parts

4.1 Answer i: Encodings

Our proposed architecture relies on pretrained monolingual embeddings for source and target languages. State-of-the-art (SOTA) transformer-based models are selected to ensure a consistent and effective representation for both sentence types.

4.1.1 Pretraining Transformer-Based Embeddings

We consider the following models for obtaining word embeddings: BERT, BART, XLM

4.1.2 Encoder Architecture

We adopt a bidirectional LSTM encoder to encode the source sentence into a dense vector representation. The bidirectional LSTM enables the capturing of long-term dependencies in sequences, providing access to both past and future context when encoding each word. The encoder is denoted as:

$$\mathbf{h}_t = \text{BiLSTM}(\mathbf{x}_t, \mathbf{h}_{t-1})$$

where $\mathbf{h}_t$ is the hidden state at time t, $\mathbf{x}_t$ is the input at time t, and BiLSTM denotes the bidirectional LSTM operation.

4.1.3 Continuous-Discrete Hybrid Latent Space

Our machine translation system utilizes a shared continuous-discrete hybrid latent space. We employ a discrete latent variable autoencoder model (Z_d) to obtain a compressed representation of sentences in the source language:

$$Z_d = \text{DiscreteAutoencoder}(\mathbf{x})$$

Similarly, a continuous latent variable autoencoder model (Z_c) is used to obtain a compressed representation of sentences in the target language:

$$Z_c = \text{ContinuousAutoencoder}(\mathbf{y})$$

The continuous representation (Z_c) captures semantic information, while the discrete space (Z_d) enables alignment. This combination allows for the modeling of both discrete and continuous aspects of the language data.

4.2 Answer ii: Encoder-Decoder Model

I would use an attentional LSTM decoder because attention provides an alignment between source and target words. This helps focus the model on relevant parts of the source when generating each target word. LSTMs have been shown to be effective seq2seq decoders. I choose this attentional sequence-to-sequence model because it provides an alignment between source and target phrases through the attention mechanism, while both the encoder and decoder can model linguistic context and long-range dependencies. This architecture has proven effective for machine translation. The attention mechanism allows the model to focus on different parts of the input sentence when generating the output sentence, which is particularly useful for machine translation tasks where the length and complexity of the input and output sentences can vary widely. I also think an attentional

encoder-decoder model like the Transformer would also be suitable, with multi-layer encoders and decoders. This provides representational power for complex translations.

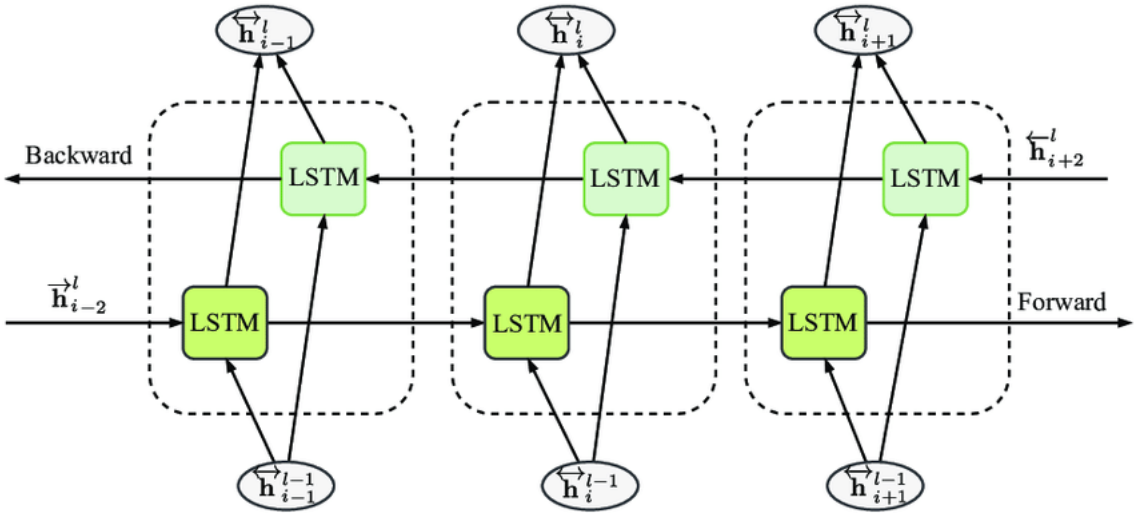

Figure 2: BiLSTM

4.2.1 Attentional LSTM Decoder

We use the following mathematical notation:

$\mathbf{h}_t^{\text{enc}}$: Hidden state of the encoder at time step t.
$\mathbf{h}_t^{\text{dec}}$: Hidden state of the decoder at time step t.
$\mathbf{c}_t$: Context vector at time step t.
$\mathbf{y}_t$: Output at time step t.
$\mathbf{x}_t$: Input at time step t.

The attention mechanism computes the context vector as a weighted sum of encoder hidden states:

$$\mathbf{c}_t = \sum_{i=1}^{T_x} \alpha_{t,i} \mathbf{h}_i^{\text{enc}}$$

where T_x is the length of the input sequence, and $\alpha_{t,i}$ are attention weights computed as:

$$\alpha_{t,i} = \frac{\exp(e_{t,i})}{\sum_{j=1}^{T_x} \exp(e_{t,j})}$$

The attention score $e_{t,i}$ is often computed using a scoring function:

$$e_{t,i} = \text{score}(\mathbf{h}_t^{\text{dec}}, \mathbf{h}_i^{\text{enc}})$$

The decoder's hidden state and context vector are then used to generate the output at time step t:

$$\mathbf{h}_t^{\text{dec}}, \mathbf{c}_t = \text{LSTM}(\mathbf{y}_{t-1}, \mathbf{h}_{t-1}^{\text{dec}}, \mathbf{c}_{t-1})$$
$$\mathbf{y}_t = \text{softmax}(\mathbf{W}_o[\mathbf{h}_t^{\text{dec}}; \mathbf{c}_t] + \mathbf{b}_o)$$

where $[\mathbf{h}_t^{\text{dec}}; \mathbf{c}_t]$ denotes concatenation, and $\mathbf{W}_o$ and $\mathbf{b}_o$ are the weight matrix and bias for the output layer.

4.2.2 Architecture Overview

The attentional sequence-to-sequence model consists of:

- An attentional LSTM decoder as described above.
- A bidirectional LSTM encoder (BiLSTM):

$$\mathbf{h}_t^{\text{enc}} = \text{BiLSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}^{\text{enc}})$$

The attention mechanism as detailed earlier.

4.3 Answer iii: Latent Variable Loss

For the continuous VAE, the loss for the latent variable z involves the Kullback-Leibler (KL) divergence between the posterior encoder distribution $q(z|x)$ and the prior $p(z)$:

$$\mathcal{L}_{\text{KL}}^{\text{continuous}} = \text{KL}(q(z|x)||p(z))$$

On the other hand, for the discrete VAE, where KL divergence is not differentiable, an alternative loss such as Jensen-Shannon (JS) divergence or energy distance is utilized. To enhance model generalization, a negative entropy term is incorporated into the KL loss, encouraging a more dispersed posterior distribution:

$$\mathcal{L}_{\text{KL}}^{\text{discrete}} = \text{JS\_Divergence}(q(z|x)||p(z)) - \lambda \cdot \text{Entropy}(q(z|x))$$

Here, λ serves as a hyperparameter that modulates the influence of the entropy term.

In the joint training of the model, the overall objective combines the Evidence Lower Bound (ELBO) losses for both source and target language reconstructions. Given the use of discrete VAEs, the reparameterization trick, specifically Gumbel-Softmax, is employed for gradient flow facilitation:

$$\begin{aligned}\mathcal{L}_{\text{joint}} = &\sum_i \Big[E_{q(y|x^{(i)};\phi_y|x)}[\log p(x|y;\theta_x|y)] \\ &-\text{KL}(q(y|x^{(i)};\phi_y|x)||p_{\text{Target}}(y))\Big] + \sum_j \left[E_{q(x|y^{(j)};\phi_x|y)}[\log p(y|x;\theta_y|x)]\right. \\ &\left.-\text{KL}(q(x|y^{(j)};\phi_x|y)||p_{\text{Source}}(x))\right]\end{aligned}$$

This formulation incorporates the ELBO losses for both source and target language reconstructions along with the KL divergence terms, ensuring a comprehensive objective for the training of the VAE model.

4.4 Answer iv: Training

For the discrete VAE, standard backpropagation is not possible due to the non-differentiable sampling operation. I would use the REINFORCE algorithm which uses Monte Carlo sampling from the encoder and rewards from the decoder to train the model. The Gumbel-Softmax relaxation is another option that provides a continuous approximation to categorical sampling for backpropagation. One approach can also be using the reparameterization trick, which involves sampling from a noise distribution and transforming the samples using a differentiable function to obtain samples from the posterior distribution. This allows for the use of backpropagation to compute gradients and update the model parameters. Another approach is to use the Gumbel-Softmax estimator or REINFORCE algorithm to approximate the gradients of the ELBO objective. Additionally, the paper[1] suggests using a stop-gradient method to approximate the gradients of the reconstruction loss, which has been shown to be more effective than other gradient estimation methods.

References

[1] He, J. X. W., & Berg-Kirkpatrick, T. (2020). A Probabilistic Formulation of Unsupervised Text Style Transfer. ICLR.

[2] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., ... & Lerchner, A. (2017). Beta-VAE: Learning basic visual concepts with a constrained variational framework. ICLR.

[3] Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., & Bengio, S. (2016). Generating sentences from a continuous space. arXiv preprint arXiv:1511.06349.

[4] Kingma, D. P., Salimans, T., & Welling, M. (2016). Improving variational inference with inverse autoregressive flow. NIPS.