Learning to Drop Out: An Adversarial Approach to Training Sequence VAEs

Anonymous Author(s)
Affiliation
Address
email

Abstract

In principle, applying variational autoencoders (VAEs) to sequential data offers a method for controlled sequence generation, manipulation, and structured representation learning. However, training sequence VAEs is challenging: autoregressive decoders can often explain the data without utilizing the latent space, known as *posterior collapse*. To mitigate this, state-of-the-art models 'weaken' the 'powerful decoder' by applying uniformly random *dropout* to the decoder input. We show theoretically that this removes *pointwise mutual information* provided by the decoder input, which is compensated for by utilizing the latent space. We then propose an *adversarial* training strategy to achieve *information-based stochastic dropout*. Compared to uniform dropout on standard text benchmark datasets, our targeted approach increases both sequence modeling performance and the information captured in the latent space.

1 Introduction

Training autoregressive models via maximum likelihood estimation (MLE) is a common strategy for representing sequential data. Such autoregressive models obtain state-of-the-art in modeling text [Brown et al., 2020, Chowdhery et al., 2022, Zhang et al., 2022], speech [Oord et al., 2018, Conneau et al., 2020], and video sequences [Babaeizadeh et al., 2018]. In this simple and intuitive approach, the joint distribution of a sequence is factorized into a product of conditional distributions, with each sequence element conditioned on its history. However, in their basic form, autoregressive models do not necessarily learn *latent variables* that reflect semantic content of the data, as often desired. As such, supplementing autoregressive models with latent variables is a promising way to enhance control in sequence generation, and enable structured representation learning.

A recent approach to latent variable modeling of sequences is to integrate autoregressive components into the *variational autoencoder* (VAE) framework [Kingma and Welling, 2013, Rezende et al., 2014]. *Sequence VAEs* offer a theoretically principled solution, but have not been widely adopted. Arguably, this is largely due to the phenomenon of *posterior collapse* [Bowman et al., 2016, Chen et al., 2017, Van Den Oord et al., 2017, Li et al., 2020]. Posterior collapse describes when a VAE's posterior probability over latent variables 'collapses' to the latent prior, rendering the latent space completely uninformative, or *independent*, of the data. Recent years have seen an abundance of research into posterior collapse [Bowman et al., 2015, Chen et al., 2017, Lucas et al., 2019, Dai et al., 2020, Fang et al., 2020, Pang et al., 2021], identifying the 'power' of autoregressive decoding as a main cause. In particular, autoregressive decoders are shown to obtain satisfactory sequence modeling performance without utilizing the latent space [Bowman et al., 2016, Chen et al., 2017]. This suggests that for sequence VAEs to progress, improved techniques are needed to understand and alleviate posterior collapse, and thereby learn informative latent structure. In this work, we propose a theoretically principled way to mitigate posterior collapse when training autoregressive sequence VAEs.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

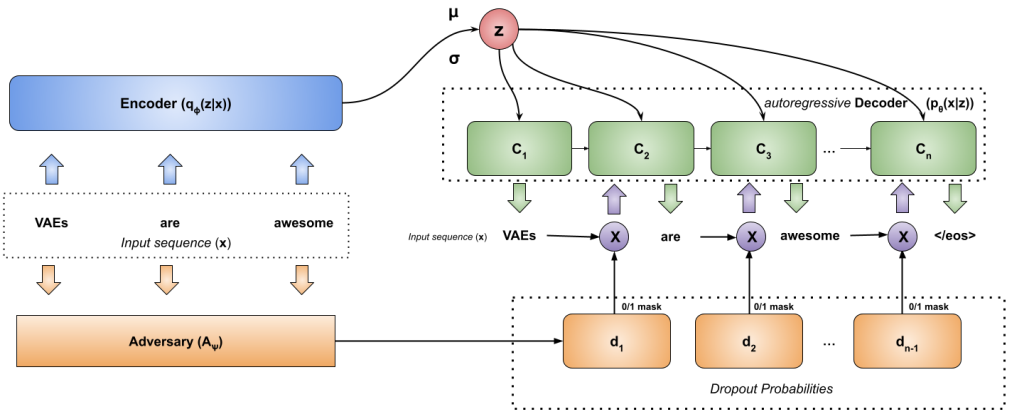

Figure 1: Adversarial training of sequence VAEs. Our proposed model comprises **an encoder**, **an autoregressive sequence decoder** (together an autoregressive VAE), and **an adversary $\mathbf{A}_\psi$**. During training, the encoder learns a representation of a full input sequence $\boldsymbol{x}$; the same sequence serves as both input (together with the hidden representation) and target to train the decoder. The adversary *learns to stochastically drop out* sequence elements that the decoder requires most, **masking (0/1)** each sequence element $\boldsymbol{x}_i$ with **probability d_i**. See Section 4.2 for details.

The intuition that autoregressive VAE decoders are too 'powerful' has led to the notion of 'weakening' them, e.g. by regularization. However, weakening an autoregressive model arbitrarily might of course harm its performance, leading to a trade off between informativeness of the latent space and the quality of sequence modeling (or density estimation performance). State-of-the-art VAEs apply variants of *dropout* [Srivastava et al., 2014] to the sequence input into the decoder: either to individual dimensions of sequence elements [Kim et al., 2018], or to mask entire sequence elements [Iyyer et al., 2015, Bowman et al., 2016]. In either case, dropout is generally applied uniformly at random, meaning that each sequence element has an equal probability of being dropped. From the intuition that different words carry different information about the next word, this work investigates whether a *non-uniform* dropout policy can achieve a better trade-off between capturing latent information and sequence modeling performance. Specifically, by stochastically dropping each sequence element according to its importance to the autoregressive decoder, our method takes a *targeted* approach to dropout relative to uniform sampling, maintaining sequence modeling performance while achieving the benefits of dropout on the latent space. To apply a non-uniform, data-dependent dropout scheme, our approach introduces an 'adversary' that *learns which elements to drop*. The proposed framework is represented in Figure 1, theoretically developed in Section 4.1 and implemented in Section 4.2.

The contributions of this paper can be summarized as follows: **(i)** We propose an adversarial approach to learning a stochastic dropout policy that mitigates posterior collapse in sequence VAEs. **(ii)** We show theoretically that dropping out sequence elements deducts *mutual information not already learned by the model*, between one sequence element and the next, and that our approach maximises this quantity, leading to a constrained minimax ELBO objective that explains an adversarial approach. **(iii)** We evaluate the proposed scheme on standard text benchmarks, showing that our approach renders an informative latent space, without trading-off but rather improving sentence modeling. **(iv)** We examine properties of the adversarial network and find that the adversary typically selects words that carry sentence semantics that fit our theoretical analysis.

2 Background

Here, we formally describe the use of autoregressive decoding within VAEs for sequential data and the problem of *posterior collapse* that often arises. For notation: $\boldsymbol{X}$ denotes a random variable, (boldface) $\boldsymbol{x}$ denotes a *sequence* that is a realization of $\boldsymbol{X}$, (subscript) $\boldsymbol{x}_i$ denotes the i-th sequence element of $\boldsymbol{x}$ and $\boldsymbol{x}_{<i}$ collectively denotes all sequence elements prior $\boldsymbol{x}_i$. z denotes a global latent vector and (superscript) $\boldsymbol{x}^{(i)}$ denotes the i-th sample from a dataset.

The task is to approximate the true probability density function $p_*(\boldsymbol{x})$ of a random sequence $\boldsymbol{X} \sim p_*(\boldsymbol{x})$ using a model $p_\theta(\boldsymbol{x})$ with parameters θ. Given a set of observations $\{\boldsymbol{x}^{(i)}\}_{i=1}^N$, θ can be estimated by *maximum likelihood estimation* (MLE):

$$\theta_{MLE} = \underset{\theta}{\operatorname{argmax}} \left[\frac{1}{N} \sum_{i=1}^{N} \log p_\theta(\boldsymbol{x}^{(i)}) \right] \tag{1}$$

If the data has latent generative factors z, modelled as $p_\theta(\boldsymbol{x}) = \int_z p_\theta(\boldsymbol{x}|z) p_\theta(z)$, Eq 1 is intractable and one can instead maximize the *evidence lower bound* (ELBO), which for any observation $\boldsymbol{x}$ is:

$$\log p(\boldsymbol{x}) \geq \log p_\theta(\boldsymbol{x}) \geq \text{ELBO}_{\theta,\phi}(\boldsymbol{x}) \triangleq \int_z q_\phi(z|\boldsymbol{x}) \log p_\theta(\boldsymbol{x}|z) - \int_z q_\phi(z|\boldsymbol{x}) \log \frac{q_\phi(z|\boldsymbol{x})}{p_\theta(z)} \tag{2}$$

where, $q_\phi(z|\boldsymbol{x})$ approximates the posterior, parameterized by ϕ, and $p_\theta(z)$ is a prior distribution over latent variables. In this work, we consider the framework of variational autoencoders (VAEs) [Kingma and Welling, 2013, Rezende et al., 2014], where the *encoder* $q_\phi(z|\boldsymbol{x})$ and *decoder* $p_\theta(\boldsymbol{x}|z)$ are parameterised by neural networks and $p_\theta(z)$ is a standard Gaussian, denoted $p_0(z)$. Further, we focus specifically on VAEs where the decoder $p_\theta(\boldsymbol{x}|z)$ is *autoregressive* [Bowman et al., 2016, Li et al., 2020, He et al., 2019], meaning $\log p_\theta(\boldsymbol{x}|z) = \sum_i \log p_\theta(x_i|\boldsymbol{x}_{<i}, z)$, and the ELBO becomes:

$$\text{ELBO}^{\mathbf{AR}}_{\theta,\phi}(\boldsymbol{x}) = \int_z q_\phi(z|\boldsymbol{x}) \sum_i \log p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i}, z) - \underbrace{\int_z q_\phi(z|\boldsymbol{x}) \log \frac{q_\phi(z|\boldsymbol{x})}{p_0(z)}}_{\text{KL}} \tag{3}$$

Posterior collapse: Autoregressive models improve density estimation [Oord et al., 2016] by explicitly modeling statistical dependencies between sequence elements. However, when used as a VAE decoder $p_\theta(\boldsymbol{x}|z)$, autoregressive models are often found to ignore z, i.e. $p(\boldsymbol{x}|z) \approx p(\boldsymbol{x})$, and the posterior is said to 'collapse' to the prior, i.e. $q_\theta(z|\boldsymbol{x}) \approx p_0(z)$, rendering z *de facto* independent of $\boldsymbol{x}$ [Bowman et al., 2016, He et al., 2019]. In terms of Eq 3, this implies $p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i}, z) \approx p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i})$. From this, posterior collapse can be considered due to the *information* about $\boldsymbol{x}_i$ provided by $\boldsymbol{x}_{<i}$ being such that each $\boldsymbol{x}_i$ is (approximately) conditionally independent of z given $\boldsymbol{x}_{<i}$, in other words z is essentially *redundant*. Based on this, we interpret the notion that autoregressive models are *too powerful* for use as a VAE decoder to mean that they allow *too much information flow*.

3 Related Work

Posterior collapse The phenomenon of posterior collapse has been observed in the context of text [Bowman et al., 2016, Yang et al., 2017], images [Chen et al., 2017, Razavi et al., 2018], videos [Babaeizadeh et al., 2018], speech [Chorowski et al., 2019] and graphs [Kipf et al., 2018]. In broad terms, previous solutions can be divided into two complementary categories: **(i)** *latent-variable-oriented* methods that focus mainly on relaxing the KL penalty; and **(ii)** *decoder-oriented* methods that regularise or 'weaken' the autoregressive decoder. Typically, both types are required in order to learn an informative latent space [Bowman et al., 2016, Chen et al., 2017, Kim et al., 2018].

Latent-variable-oriented solutions Bowman et al. [2016] suggest *KL annealing*, a technique that introduces the KL term gradually into training according to a predefined schedule. *Free bits* [Kingma et al., 2016] prevent penalizing of the KL term if its magnitude is below a predefined threshold. Van Den Oord et al. [2017] use a *discrete* VAE, avoiding posterior collapse by design. Fu et al. [2019] apply a *cyclical* form of KL annealing. Razavi et al. [2018] constrain the variational family of the posterior distribution (encoder), preventing it from closely approximating the prior and so holding the KL term away from zero. *Lagging inference networks* [He et al., 2019] 'aggressively' optimize the encoder before each decoder update. A similar procedure is followed by *Semi-Amortized VAEs* [Kim et al., 2018]. *Generative skip models* [Dieng et al., 2019] introduce skip connections to create a more explicit link between the latent variables and the likelihood function. Sinha and Dieng [2021] propose a *consistency* regularizer by minimizing the KL divergence between the posterior approximations of an observation and a random transformation of it.

Decoder-oriented solutions Bowman et al. [2016] apply *word dropout* [Iyyer et al., 2015] to uniformly drop words during autoregressive decoding. Other methods [Kim et al., 2018, He et al., 2019] apply parameter dropout [Srivastava et al., 2014] to word embeddings. Chen et al. [2017],

Semeniuta et al. [2017], Yang et al. [2017] constrain the receptive field of the decoder, limiting the window of autoregression, however, this is not readily applicable to RNN architectures given their unbounded receptive field. Our proposed adversarial method falls into this category as an extension of word dropout, which is then subsumed as a special case of adversarial word dropout (Section 4.2).

Non-uniform dropout Previous work has recognized the benefit of adapting the dropout rate across different architectural components, though in entirely different contexts [Kingma et al., 2015, Gal et al., 2017, Achille and Soatto, 2018]. These works introduce different approaches to regularizing the magnitude of weights or activations (groups of weights). Elements that are deemed *irrelevant* during training are then dropped. For instance, variational dropout [Kingma et al., 2015] has been used to prune weights in deep neural networks [Molchanov et al., 2017]. A key difference to our approach is that instead of a regularization term, we employ an adversary that selects elements for dropout during training based on their information content not yet learned by the model.

Exposure bias The method by which the autoregressive decoder is trained, known as *teacher forcing*, bears a connection to *exposure bias*, which refers to the gap between training and inference [Ranzato et al., 2015]. Namely, in language generation, a trained model generates sequences at test time without access to the ground truth history that was accessible during training. Since the model was not trained to continue its predictions, this can lead to error accumulation [Ranzato et al., 2015]. The most common approach to tackle exposure bias is to switch between conditioning on ground truth and model predictions, with the latter being preferred towards the end of training [Daumé et al., 2009, Bengio et al., 2015, Ranzato et al., 2015]. In preliminary studies, we implemented *scheduled sampling* [Bengio et al., 2015] but did not find it to outperform even uniform dropout.

4 Adversarial Word Dropout (AWD)

4.1 Theoretical Basis

As a precursor to our adversarial approach, we first derive the effect of stochastic dropout on the ELBO under the autoregressive assumption (Equation 3).

Word dropout: Our interpretation of posterior collapse at the end of Section 2 suggests that to 'weaken the decoder' one should *restrict the available information*, such that z is no longer redundant and must capture some of the restricted information. Word dropout [Bowman et al., 2016] can be seen to do this by stochastically masking the previous word $\boldsymbol{x}_{i-1}$ with probability d_i, which substitutes $p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i}, z)$ with $p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i-1}, z)$ in Equation 3, i.e.:

$$\text{ELBO}^{\mathbf{WD}}_{\theta,\phi}(\boldsymbol{x}) = \int_z q_\phi(z|\boldsymbol{x})\Big\{\sum_i (1-d_i)\log p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i}, z) + d_i \log p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i-1}, z)\Big\} - \text{KL} \tag{4}$$

$$= \int_z q_\phi(z|\boldsymbol{x})\Big\{\sum_i \log p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i}, z) - \sum_i d_i \underbrace{\frac{p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{i-1}, \boldsymbol{x}_{<i-1}, z)}{p_\theta(\boldsymbol{x}_i|\boldsymbol{x}_{<i-1}, z)}}_{\text{PMI}(\boldsymbol{x}_i, \boldsymbol{x}_{i-1}|\boldsymbol{x}_{<i-1}, z)}\Big\} - \text{KL} \tag{5}$$

Equation 5 shows the effect (in expectation) of applying word dropout when maximising the ELBO with an autoregressive decoder in Eq 3 (of which 'KL' denotes the last term). In effect, a weighted sum is introduced with weights given by dropout probabilities d_i and components that define conditional *point-wise mutual information* (PMI), where classic point-wise mutual information between two variables is defined as $\text{PMI}(\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{p(\boldsymbol{x}_1|\boldsymbol{x}_2)}{p(\boldsymbol{x}_1)}$. Each PMI term in Eq 5 captures the information that one sequence element $\boldsymbol{x}_{i-1}$ has regarding the next $\boldsymbol{x}_i$, *over and above* any information from earlier elements $\boldsymbol{x}_{<i-1}$ and the latent state z. Since the dropout-adapted ELBO (Eq 5) is maximised w.r.t. θ and ϕ, the weighted PMI term is minimised, which is only achievable by *increasing the information learned by the model*: extracted from $\boldsymbol{x}_{<i-1}$ or captured in z. Interestingly this shows that word dropout quite literally follows the earlier intuition and 'weakens' the decoder by restricting *information*.

The minimax objective: Under uniform dropout, all d_i are equal over a sequence, and using dropout to 'push information into z' is applied evenly. However, some sequence elements may hold more information than others, e.g. in language, given syntactic rules and a summary of the previous

text, some words may be highly predictable without specifically knowing which word preceded them, whereas others may depend heavily on their predecessor despite the other information. This suggests that information content may be non-uniform and so applying dropout non-uniformly may be more appropriate. We, therefore, propose *targeted* dropout of elements according to their 'incremental' information content defined by the PMI terms, i.e. that not yet learned by the model. Further, since such incremental information content is not explicitly computed and will vary over training as the model learns, we train an adaptive dropout schedule to continually maximise the information dropped, and so minimize Eq 5, with respect to d_i. Since Eq 5 is maximised with respect to all other parameters, this leads to a minimax objective.

We introduce the two 'players' in the proposed minimax setting: $\text{VAE}_{\phi,\theta}$, trained to maximize the ELBO, and the adversary A_ψ trained to do the opposite by determining each probability d_i of dropping out the previous sequence element $\boldsymbol{x}_{i-1}$, preventing it from helping to predict its successor $\boldsymbol{x}_i$. The full objective for an observed sequence is given by

$$\max_{\phi,\theta} \min_{\psi} \mathbb{E}_{\boldsymbol{x}\sim p_*(\boldsymbol{x})} \left[\mathcal{L}_{\psi,\phi,\theta}(\boldsymbol{x}) + \mathcal{R}_\psi(\boldsymbol{x})\right] \tag{6}$$

where $\mathcal{R}_\psi(\boldsymbol{x})$ is a regularization term explained below (Section 4.2) and

$$\mathcal{L}_{\psi,\phi,\theta} = \mathbb{E}_{q_\phi(z|\boldsymbol{x})}\left[\sum_i \log p_\theta(\boldsymbol{x}_i|\text{mask}_{K,\psi}(\boldsymbol{x}_{i-1};\boldsymbol{x}), \boldsymbol{x}_{<i-1}, z)\right] - \text{KL}(q_\phi(z|\boldsymbol{x})||p_0(z)) \tag{7}$$

$\mathcal{L}_{\psi,\phi,\theta}(\boldsymbol{x})$ is a modified version of the ELBO in Equation 5 with the additional parameters ψ of the adversary A_ψ. The adversary manifests via the masking operator $\text{mask}_{K,\psi}(\cdot\,;\boldsymbol{x})$, conditioned on the entire input sequence $\boldsymbol{x}$. Since unconstrained minimisation of Eq 7 w.r.t ψ would cause all elements to be dropped out, a constraint K is introduced such that exactly K elements of $\boldsymbol{x}$ are dropped during autoregressive decoding. In practice, K is sequence length-dependent, treated as a hyperparameter that controls the level of adversarial regularization. The adversary plays no role at test time.

4.2 Implementation

A high-level overview of our framework is shown in Figure 1. An encoder maps the input sequence into a hidden representation to produce the parameters of $q_\phi(z|x)$, from which z is sampled using the 'reparametrization trick' [Kingma and Welling, 2013, Rezende et al., 2014]. We implement the decoder using a unidirectional LSTM [Hochreiter and Schmidhuber, 1997] for consistency with the RNN-VAE network of Bowman et al. [2016], but other distributional approximators can be used that need not be autoregressive, e.g. attention-based transformer networks [Vaswani et al., 2017] are a promising choice to explore in future work. The autoregressive decoder is also implemented using a unidirectional recurrent network. Inspired by the recent work of [Dieng et al., 2019], we choose a *Double-LSTM* recurrent unit that elucidates the *skip-connections* to promote higher latent information content flow (see Appendix **??** for details).

As described in Section 4.1, the main innovation of our proposed method is the *adversarial* training procedure: instead of dropping out decoder inputs uniformly at random, we introduce a trainable adversary to drop words most important for the VAE to reconstruct the original sequence. Below we describe implementation details of the adversary, specifying the relevant components in Figure 1. A more comprehensive figure specific to the implemented architecture is provided in Appendix **??**. Note that the theoretical framework in Section 4.1 permits many alternative approaches for generating per word dropout probabilities d_i, such as attention mechanisms mentioned above. We leave further refinement of the adversarial architecture to future work.

Producing dropout scores To determine which elements to drop out of a sequence, the adversary A_ψ samples a *score* $s_i \in \mathbb{R}$ for each sequence element $\boldsymbol{x}_i$. Sequence elements with the K smallest scores are dropped. The scores are sampled from distributions $p_\psi(s_i|\boldsymbol{x})$ conditioned on the input sequence $\boldsymbol{x}$, modelled as a series of Gaussians with mean (μ) and variance (σ) parameterised by the outputs of a unidirectional LSTM. Scores are generated using the reparameterization trick, reducing gradient variance [Kingma and Welling, 2013]. Thus, $p_\psi(s_i|\boldsymbol{x})$ is described as

$$s_i|\boldsymbol{x} \sim \mathcal{N}(\mu_i, \sigma_i; \boldsymbol{x}, \psi) \qquad \text{where,} \quad [\mu_i, \sigma_i] = \text{Linear}_\psi(\text{LSTM}_{i,\psi}(\boldsymbol{x}))$$

The stochastic generation of scores gives the adversary an 'exploratory' capability during training, e.g. preventing it becoming 'stuck' on a set of sequence elements with high information content

that the model cannot learn, meaning that their PMI terms (in Eq 5) remain large and the adversary repeatedly selects them to minimise the ELBO.

Top-K word selection Based on the sampled scores $\boldsymbol{s} = \{s_i\}_{i=1}^{T}$, the subset of K words with the smallest scores are masked during decoding. To estimate gradients of the objective with respect to the parameters ψ, we use a stochastic softmax trick from Paulus et al. [2020]: during training, the stochastic subset selected in the forward pass is relaxed to admit a (biased) reparameterization gradient in the backward pass. In our experiments, we use a straight-through variant of the trick for our method to resemble word dropout, i.e. to produce discrete values $\in \{0, 1\}$. The relaxation is only used in the backward pass to compute the gradient estimator.

Gradient reversal The final component of the adversarial network is the *gradient reversal* layer [Ganin et al., 2016]. In the forward pass, the layer performs no transformation to the input; in the backward pass, the gradients are negated. Gradient reversal offers a computationally simple method to ensure that the parameters ψ of A_ψ are updated such that ELBO is minimized. If $\boldsymbol{I}$ is an identity matrix, the 'pseudo-function' of gradient reversal can be described as

$$f(\boldsymbol{x}) = \boldsymbol{x} \qquad \text{(forward pass)} \qquad\qquad \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} = -\boldsymbol{I} \qquad \text{(backward pass)}$$

4.3 Optimization challenges

Since our adversarial dropout network is fully differentiable, it is readily optimized by gradient methods, such as backpropagation. However, the magnitudes of the word dropout scores $\boldsymbol{s}$ sampled from distributions parameterized by an LSTM are unconstrained. Controlling the magnitude of the scores was found to be important for the adversary to maintain its exploratory capability (discussed above). We therefore add a KL-divergence between the distribution of each score $p(s_i|\boldsymbol{x}_i)$ and a standard Gaussian $p_0(s_i) = \mathcal{N}(s_i; 0, 1)$ to 'regularize' scores in Eq (6), subject to a scalar $\lambda > 0$:

$$\mathcal{R}_\psi(\boldsymbol{x}) = \lambda \sum_i \mathrm{KL}(p_\psi(s_i|\boldsymbol{x}) \,||\, p_0(s_i)) \tag{8}$$

Adversarial vs. random word dropout Standard word dropout [Iyyer et al., 2015, Bowman et al., 2016] can be seen as a special case of adversarial word dropout. By setting λ sufficiently high, the regularization term in Eq 8 can be made to dominate such that $p_\psi(s_i|\boldsymbol{x}) \approx p_0(s_i)$ and all scores s_i are effectively sampled from a standard Gaussian $p_0(\boldsymbol{s}_i)$ and words are dropped approximately uniformly. On the other hand, for small values of λ, A_ψ will learn which words $\mathrm{VAE}_{\phi,\theta}$ relies upon most to accurately decode the sequence and target those for drop out. As a result, whilst adding another hyperparameter to calibrate, λ offers a simple way to moderate the difference between adversarial word dropout and standard uniform word dropout, which we consider in our experiments.

5 Experiments

Here we present the results of our experiments that: **(i)** demonstrate that a sequence VAE trained with adversarial word dropout (AWD) outperforms similar models of the same class, achieving either competitive or improved sentence modeling performance and/or informativeness of the latent space; **(ii)** examine the contributions and behaviour of the adversarial network's components and hyperparameters, and **(iii)** qualitatively study the trained adversary and VAE.

Datasets We conducted experiments on 3 different datasets: *Yahoo* questions and answers [Yang et al., 2017], *Yelp* reviews [Yang et al., 2017] and downsampled Stanford Natural Language Inference (*SNLI*) corpus [Bowman et al., 2015, Li et al., 2019]. Yahoo and Yelp datasets were used in many previous works [Yang et al., 2017, Kim et al., 2018, He et al., 2019, Fu et al., 2019, Dieng et al., 2019] hence were used to benchmark our proposed method against comparable related works and also against standard word dropout. Yahoo and Yelp contain sentences with average lengths of 78 and 96 words respectively. The SNLI sentences are much shorter with an average length of 9 words and are more suitable for qualitative studies. All three datasets contain 100K sentences in the training set, 10K in the validation set, and 10K in the test set.

	PPL ↓	-ELBO ↓	KL ↑	MI ↑	PPL ↓	-ELBO ↓	KL ↑	MI ↑
Existing sequence VAEs		**Yahoo**				**Yelp**		
CNN [Yang et al., 2017]	63.90	-	10.0	-	41.1	-	**7.6**	-
Lagging [He et al., 2019]	-	**328.4** (0.2)	5.7 (0.7)	2.90	-	357.2 (0.1)	3.8 (0.2)	2.4
SA [Kim et al., 2018]	60.40	-	7.19	-	-	-	-	-
Skip [Dieng et al., 2019]	60.90	330.3	**15.05**	7.47	-	-	-	-
FBP [Li et al., 2019]	59.51	330.3	15.02	-	-	-	-	-
Our sequence VAE								
unregularized	60.30	328.8 (0.2)	4.2 (0.2)	3.14	40.1	356.4 (0.2)	2.3 (0.2)	1.0
+ word dropout [0.4]	59.55	329.5 (0.4)	14.4 (0.4)	**13.6**	38.5	**354.2** (0.3)	5.9 (0.4)	4.9
+ AWD (*ours*) [0.3]	**59.05**	**328.4** (0.3)	14.4 (0.4)	**13.6**	**38.2**	**354.2** (0.3)	**6.5** (0.4)	**5.8**

Table 1: **Results of text modeling on the Yahoo and Yelp datasets.** Standard deviations are provided in the brackets. Squared bracket contains the dropout rate DR. PPL – perplexity; ELBO – evidence lower bound; KL - in Eq (7); MI – mutual information $I(\boldsymbol{X}; Z)$ from Section 2.

Experimental setup We (re-)implement the standard VAE, a VAE with uniform dropout and a VAE with our proposed dropout, each with the same hyperparameter configuration of previous comparable works [Bowman et al., 2015, Kim et al., 2018, He et al., 2019], except for the differences specified below applied fairly to each implementation. We use SGD with a learning rate of 1 and exponential decay of 0.96, which was found to improve training stability compared to decay based on validation performance [Li and Arora, 2019]. We increase the hidden state size of the decoder LSTM from 1024 to 2048 to improve overall performance, except for the SNLI dataset, where this offered no improvement. We apply Polyak averaging [Polyak and Juditsky, 1992] with a coefficient of 0.9995, slightly improving validation performance. We found *KL annealing* [Bowman et al., 2016] necessary, but apply no other techniques to improve performance proposed in previous works (see Section 3). We apply early stopping based on when the ELBO of the validation set reaches a maximum. Standard deviations are computed over 5 random seeds. All experiments are performed on a 12GB Nvidia TitanXP GPU with an average run time of 4 hours for Yelp and Yahoo and 1 hour for SNLI.

Metrics Overall model performance is assessed based on two main aspects: **(i)** *sentence modeling* – measured with respect to ELBO and perplexity (PPL). ELBO and PPL quantify the ability of the trained decoder to recognise or generate natural language, reflecting the quality of density estimation; **(ii)** *latent space informativeness* – measured with respect to the KL term in Eq (7) and mutual information $I(\boldsymbol{X}; Z)$ between observed and latent variables. $I(\boldsymbol{X}; Z)$ is computed using the procedure of Hoffman and Johnson [2016], as implemented by Dieng et al. [2019]. As in comparable works [Yang et al., 2017, Kim et al., 2018, He et al., 2019, Dieng et al., 2019, Li et al., 2019], hyperparameters are set based on a balanced assessment of these metrics.

Adversary hyperparameters The number K of elements dropped out from a particular sequence is calculated from a fixed *global dropout rate* R and the sequence length T, as *round*$(R \times T)$ where *round* computes the closest integer. We compare validation results for $R \in \{0.2, 0.3, 0.4, 0.5\}$, and use $R = 0.3$ in our final model; meaning that $\approx 30\%$ of each sentence is dropped.

To set λ in Eq 8, we compare $\lambda \in \{0.001, 0.01, 0.1, 1, 5, 10\}$ and find $\lambda = 1$ to work consistently well across all datasets. Below we provide a study on the effects of λ. The adversary is trained with the Adam optimizer [Kingma and Ba, 2014] with the learning rate selected from a hyperparameter search over $\{0.0001, 0.001, 0.1, 1\}$: 0.01 for SNLI and 0.0001 for Yahoo and Yelp.

5.1 Quantitative analysis

Table 1 compares **(i)** a VAE trained with no dropout; **(ii)** a VAE trained with uniform word dropout [Bowman et al., 2016] with dropout rate $R = 0.4$; **(iii)** a VAE trained with adversarial word dropout ($R = 0.3$); and **(iv)** previously reported results for comparable models (Section 3).

The main observation from these results is that our proposed adversarial approach achieves a better balance between learning an informative latent space (high MI) and sequence modeling (PPL) compared to uniform word dropout. Indeed, training with adversarial dropout achieves either comparable or improved performance on all metrics, suggesting that the model not only better captures the data distribution $p(\boldsymbol{x})$, but does so by better reflecting its latent structure. Our method

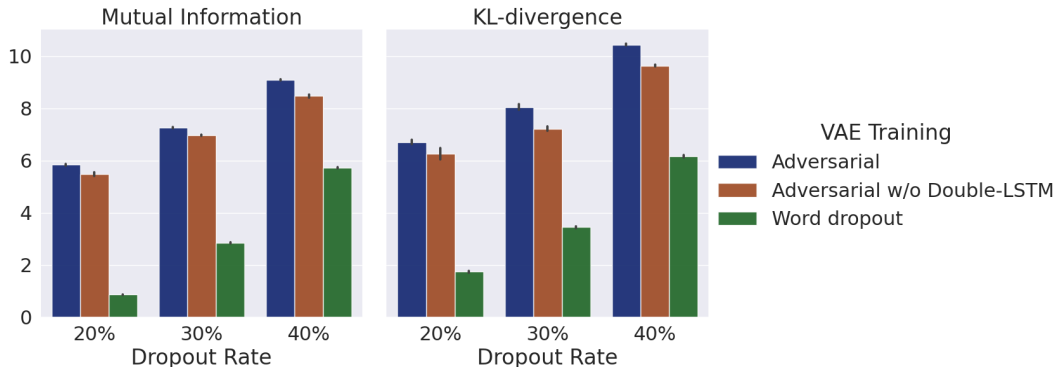

Figure 2: Adversarial vs uniform word dropout (SNLI), with ablation of the Double-LSTM.

also compares favourably to previous models, consistently achieving an improved balance between sentence modeling performance (e.g. PPL) and an informative latent space (e.g. MI).

We note also that the vanilla VAE does not obtain a lower perplexity than either VAE with word dropout, as might be anticipated if dropout were an arbitrary 'regularisation' method that may improve the latent space but at a cost to sequence modeling performance. This supports our theoretical analysis indicating that word dropout is not a typical 'regularizer' (Section 4.1), rather that it disrupts the flow of available information during training of the autoregressive decoder, forcing it to compensate by storing information in the latent space.

Figure 2 compares adversarial and uniform word dropout, varying the dropout rate on the SNLI dataset. For any given dropout rate, adversarial dropout learns a more informative latent space under both metrics. The figure also ablates the benefit of the Double-LSTM.

The role of λ The hyperparameter λ can be seen to determine the *exploration-exploitation trade-off* of the adversary. As shown in Figure 3, for small values of λ, the magnitudes and standard deviations of word dropout scores grow large, causing the distribution of dropped words to become concentrated. Figure 3 aligns with our findings that $\lambda = 1$ serves as a useful choice. For large values of λ, scores become very small with low variance and the adversary converges towards uniform word dropout.

Qualitative analysis We obtain further insight into what the adversary learns by analysing the word dropout scores for different sentences. Table 2 shows that the adversary applies lower dropout probabilities to less informative words such as 'unknown' tokens that replace all out-of-dictionary words, and so offer little information about the next word. Depending on the data semantics, the adversary selects different types of words: for SNLI, verbs tend to be picked that explain the activity, e.g. *working* and *participating*; for Yahoo, words are identified that carry question semantics, e.g. *what*, *how*, *if*, and *when*. We also show that adversarial training learns a useful generative model with meaningful latent space by interpolating between sentences (Table 3).

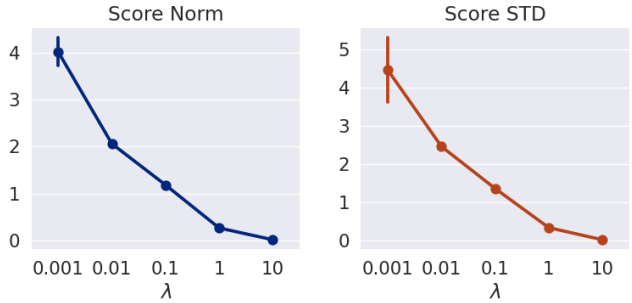

Figure 3: The role of λ (Yahoo). Mean ℓ_1-norm and standard deviation of dropout score vectors ($\boldsymbol{s}$).

<sos> the person giving the demonstration is a man <eos>

<sos> a dog participating in a race while wearing the number 6 . <eos>

<sos> three women are talking about their matching wigs and silver leggings . <eos>

<sos> is there any truth to the rumor that the host of the _unk on _unk , greg _unk is dead ? _unk has been down and there is no new information on the web about the future of the site or the _unk , greg _unk . what 's going on ? how could _unk not comment on the past us election (nov. 7th) nor the iranian nuclear program ? something must be wrong , where is this guy ? internet chat rooms have noted that _unk was injured in a fly fishing accident earlier this month . details are unknown . can anyone clear this up ? <eos>

<sos> is it okay if i mind _unk with my goldfish ? yes , but be careful , you may be overwhelmed when you connect to such a clearly superior intellect ... <eos>

Table 2: Analysis of the Adversary. For selected SNLI *(upper)* and Yahoo *(lower)* sentences, word dropout scores are from a trained adversary and normalized per sentence. Darker colouring indicates a higher dropout probability. Boxed words are those selected to be dropped.
<sos> = start of the sequence token; <eos> = end of the sequence token; _unk = unknown token.

I 'm not sure what all the hype is about . i 've been here a few times and it 's just ok . nothing special . I would n't go out of my way to come here .

I 've been here a few times and it 's always been good . the food is good , but the service is not so great .
I 've been here a few times and it 's always been good . i 've had the chicken and waffles and the service was good .
Great place to go for a quick bite to eat . the food is great and the service is great . i have been here a few times and have never been disappointed .

Great food , great service , and great service . i 've been here a few times and have never been disappointed

Table 3: Sentence interpolation (Yelp dataset). Representations of two sentences *(top, bottom)* are obtained by feeding them through an adversarially trained VAE encoder. Three linearly interpolated representations are passed to the VAE decoder and sentences generated by greedy sampling *(middle)*.

6 Conclusions

In this work, we address the phenomenon of *posterior collapse* that occurs in autoregressive variational autoencoders (VAEs) used for sequential data. To mitigate posterior collapse, we propose a novel adversarial approach that *learns to drop words based on their information content*, leading to an improvement in both the learned latent structure and sequence modeling performance. More fundamentally, we theoretically derive the effect of word dropout to show that, during training, it removes *incremental* information that each word provides about the next above that available from earlier words or the latent variable z. The only way for the model to compensate and minimise the incremental information lost is to *learn more information in the latent space*. We believe this finding provides novel and interesting insight as a means of *manipulating information* within a hierarchical latent variable model since dropout is used to 'push' specific information into the latent variable.

Sequence VAEs are a promising framework that, beyond purely autoregressive models [Brown et al., 2020], hold the prospect of controlled sequence generation. Improvements to such general methods may inevitably be used for good or ill, from the generation of targeted fake news on the one hand to the possibility of personalised human-computer interactions in, say, the medical domain on the other. In future work, we plan to explore alternative implementation options, particularly of the adversary, and extend adversarial dropout to other domains, such as images or speech.

References

Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE transactions on pattern analysis and machine intelligence*, 40 (12):2897–2905, 2018.

Mohammad Babaeizadeh, Chelsea Finn, Dumitru Erhan, Roy H Campbell, and Sergey Levine. Stochastic variational video prediction. In *International Conference on Learning Representations*, 2018.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Empirical Methods in Natural Language Processing*, 2015.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In *Conference on Computational Natural Language Learning*, 2016.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. *ICLR*, 2017.

Jan Chorowski, Ron J Weiss, Samy Bengio, and Aäron van den Oord. Unsupervised speech representation learning using wavenet autoencoders. *IEEE/ACM transactions on audio, speech, and language processing*, 27(12):2041–2053, 2019.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. Unsupervised cross-lingual representation learning for speech recognition. *arXiv preprint arXiv:2006.13979*, 2020.

Bin Dai, Ziyu Wang, and David Wipf. The usual suspects? reassessing blame for vae posterior collapse. In *International Conference on Machine Learning*, pages 2313–2322. PMLR, 2020.

Hal Daumé, John Langford, and Daniel Marcu. Search-based structured prediction. *Machine learning*, 75(3):297–325, 2009.

Adji B Dieng, Yoon Kim, Alexander M Rush, and David M Blei. Avoiding latent variable collapse with generative skip models. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2397–2405. PMLR, 2019.

Xianghong Fang, Haoli Bai, Zenglin Xu, Michael Lyu, and Irwin King. Discrete auto-regressive variational attention models for text modeling. *arXiv:2004.09764 [cs, stat]*, Oct 2020. URL `http://arxiv.org/abs/2004.09764`. arXiv: 2004.09764.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. Cyclical annealing schedule: A simple approach to mitigating KL vanishing. 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 240–250, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1021. URL `https://www.aclweb.org/anthology/N19-1021`.

Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout. *arXiv preprint arXiv:1705.07832*, 2017.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Junxian He, Daniel Spokoyny, Graham Neubig, and Taylor Berg-Kirkpatrick. Lagging inference networks and posterior collapse in variational autoencoders. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=rylDfnCqF7`.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NIPS*, volume 1, page 2, 2016.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 1681–1691, 2015.

Yoon Kim, Sam Wiseman, Andrew Miller, David Sontag, and Alexander Rush. Semi-amortized variational autoencoders. In *International Conference on Machine Learning*, pages 2678–2687, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Diederik P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. *arXiv preprint arXiv:1506.02557*, 2015.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems*, pages 4743–4751, 2016.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. *arXiv preprint arXiv:1802.04687*, 2018.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. A surprisingly effective fix for deep latent variable modeling of text. *arXiv preprint arXiv:1909.00868*, 2019.

Ruizhe Li, Xiao Li, Guanyi Chen, and Chenghua Lin. Improving variational autoencoder for text modelling with timestep-wise regularisation. *arXiv preprint arXiv:2011.01136*, 2020.

Zhiyuan Li and Sanjeev Arora. An exponential learning rate schedule for deep learning. *arXiv preprint arXiv:1910.07454*, 2019.

James Lucas, George Tucker, Roger B Grosse, and Mohammad Norouzi. Don't blame the elbo! a linear vae perspective on posterior collapse. In *Advances in Neural Information Processing Systems*, pages 9408–9418, 2019.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational dropout sparsifies deep neural networks. In *International Conference on Machine Learning*, pages 2498–2507. PMLR, 2017.

Aaron Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pages 3918–3926. PMLR, 2018.

Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with pixelcnn decoders. *arXiv preprint arXiv:1606.05328*, 2016.

Bo Pang, Erik Nijkamp, Tian Han, and Ying Nian Wu. Generative text modeling through short run inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, page 1156‘Ă‘Ş1165, Online, Apr 2021. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2021.eacl-main.98`.

Max B Paulus, Dami Choi, Daniel Tarlow, Andreas Krause, and Chris J Maddison. Gradient estimation with stochastic softmax tricks. *arXiv preprint arXiv:2006.08063*, 2020.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

Ali Razavi, Aaron van den Oord, Ben Poole, and Oriol Vinyals. Preventing posterior collapse with delta-vaes. In *International Conference on Learning Representations*, 2018.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. A hybrid convolutional variational autoencoder for text generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 627–637, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

Samarth Sinha and Adji Bousso Dieng. Consistency regularization for variational auto-encoders. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 12943–12954. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper/2021/file/6c19e0a6da12dc02239312f151072ddd-Paper.pdf`.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pages 6306–6315, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] (Section 4.2)
   (b) Did you describe the limitations of your work? [Yes] (Section 6)
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] (Section 6)
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] (All Sections)

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] (Section 4.1)
   (b) Did you include complete proofs of all theoretical results? [Yes] (Section 4.1)
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] (Supplementary)
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] (Section 5)
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] (Table 1)
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] (Section 5)
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] (Section 5)
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]