# Eval all, trust a few, do wrong to none: Comparing sentence generation models

**Ondřej Cífka**[*]
Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
cifka@matfyz.cz

**Aliaksei Severyn**
Google Inc.
severyn@google.com

**Enrique Alfonseca**
Google Inc.
ealfonseca@google.com

**Katja Filippova**
Google Inc.
katjaf@google.com

## Abstract

In this paper, we study recent neural generative models for text generation related to variational autoencoders. These models employ various techniques to match the posterior and prior distributions, which is important to ensure a high sample quality and a low reconstruction error. In our study, we follow a rigorous evaluation protocol using a large set of previously used and novel automatic metrics and human evaluation of both generated samples and reconstructions. We hope that it will become the new evaluation standard when comparing neural generative models for text.

## 1 Introduction

While automatic natural language generation (NLG), in particular from structured data, has had a long tradition (Reiter and Dale, 2000), the recent advances in deep learning have given it a new impetus. In parallel to a massive number of deep generative models for creating realistic images, a fair number of papers have introduced probabilistic generative models of text (Zhao et al., 2017a; Hu et al., 2017, inter alia) which are claimed to produce fluent and meaningful samples from a continuous vector representation. Similar to research focused on image generation, related but distinct text generation tasks for such models include:

(1) sentence reconstruction – given a natural language sentence, can we encode it into a fixed-length vector and then reconstruct it from that representation?

(2) unconditional sentence generation – can we generate fluent sentences that follow the distribution of sentences in natural language?

(3) conditional sentence generation – given a content and/or style representation, can we generate a sentence expressing that content and exhibiting the desired stylistic properties?

While tasks (1) and (2) may not have obvious applications, they are important for assessing the properties of the learned representations and their usefulness for other tasks, including (3). For example, autoencoder-based models have been proposed (Hu et al., 2017; Zhao et al., 2017a) for learning disentangled representations of style and content from unaligned data. One possible application of such autoencoders is modifying the style of a sentence by manipulating the style representation, but this is only possible if the model can encode and generate accurately, i.e. has a low reconstruction error.

Partly due to the difficulty of evaluating text generation directly, recent studies on autoencoders for text (Bowman et al., 2016; Hu et al., 2017) have mostly focused on applying them to tasks such as language modeling and classification. With the exception of Zhao et al. (2017a), these studies do not consider the reconstruction task. Bowman et al. (2016) report negative results of variational autoencoders on language modeling, which suggests that the reconstruction error of these models will be high.

[*]Work done during an internship at Google Zurich.

The lack of evaluation standards resulted in fierce debates around the experimental setup of some of the most novel neural-based text generation studies, to the point that their utility has been questioned.[1] This is unfortunate because neural generative models for text do hold real promise for NLG, as the progress in MT over the past few years has clearly demonstrated.

In this paper, we strive to address the methodological issues with the current neural text generation research and also close some gaps by answering a few natural questions to the studies already published. We focus on neural generative models from the autoencoder family and their performance on tasks (1) and (2), because we feel that this is an area that hasn't been sufficiently explored and deserves a proper treatment before one moves on to more complex setups.

In particular, our contributions are as follows:

1. We focus on several most recent autoencoder models for sentence generation, namely plain (AE), variational (VAE) and adversarially regularized (ARAE) autoencoders (Kingma and Welling, 2013; Bowman et al., 2016; Zhao et al., 2017a), as well as adversarial autoencoders (AAE, Makhzani et al., 2015), and compare them on equal footing.

2. We study the effects of alternative techniques for regularizing autoencoders for text, namely latent code normalization, injecting noise into the latent representation, and RNN dropout.

3. We show that these simple techniques are sufficient for training an autoencoder which is comparable to state-of-the-art models for unconditional text generation while outperforming them in terms of reconstruction accuracy.

4. We rigorously evaluate different variants of autoencoder models with humans as well as compute a rich set of automatic metrics on both generated samples and reconstructions, which is missing in the previous work.

5. In particular, we introduce a novel technique for automatically measuring the quality of generated texts – *Fréchet InferSent Distance* (inspired by the recent work on image generation by Heusel et al., 2017).

[1] See the discussions around the posts by Yoav Goldberg from June 2017: https://medium.com/@yoav.goldberg.

## 2 Related work

Since the introduction of VAEs by Kingma and Welling (2013) and its many successful applications in computer vision, the first study of VAEs for text generation was performed by Bowman et al. (2016). While demonstrating that VAEs can be a viable way to train unconditional generative models for text, the authors show that training a VAE with an LSTM (**?**) decoder leads to an issue where it tends to ignore the latent code completely and hence collapse to a language model. To alleviate this issue, the authors applied two moderately successful tricks: input dropout and KL term annealing. While demonstrating their model can generate natural-looking samples, the reconstruction performance is omitted from the discussion, which is important as it indicates how well the encoder generalizes and structures the latent space.

To address some of the issues of training VAE models for text discussed by Bowman et al. (2016), Semeniuta et al. (2017) propose a hybrid architecture composed of a convolutional encoder and a decoder composed of a deconvolutional and an autoregressive layer (LSTM or ByteNet (Kalchbrenner et al., 2016)). This model is shown to better handle longer sequences and more importantly, it allows for a better control over the KL term. The latter ensures that the latent vector is actually useful and used by the decoder. Additionally, similar to findings in Chen et al. (2016) and Yang et al. (2017), explicit control over the autoregressive power of the decoder, e.g., by using a ByteNet decoder with a smaller receptive field, helps to alleviate this issue. In this work we employ a standard LSTM encoder/decoder architecture, whereas our primary focus is on various mechanisms to match posterior and prior distributions and its effects on structuring the latent space.

The original VAE objective includes a KL penalty term whose goal is to match the approximate posterior with a prior. This regularizes (smooths out) the latent space, ensuring that it is possible to generate meaningful samples from any point from the prior. Instead of using a conventional KL penalty, Makhzani et al. (2015) propose to use a GAN discriminator to match the aggregate approximate posterior with the prior. Bous-

quet et al. (2017) provide a proof that this in effect corresponds to minimizing a Wasserstein distance in the primal between the data and generated distributions. Zhao et al. (2017a) attack the problem from a different angle by using a GAN to instead learn a powerful prior that matches the aggregated posterior. Thus, during generation, the latent vectors are sampled from the GAN generator instead of being drawn directly from an imposed prior.

Hu et al. (2017) propose a conditional VAE model for text where a discriminator is used to impose desired attributes on generated samples and disentangle them from the latent representation produced by the encoder. To enable back-propagation from the discriminator, the recurrent decoder is made fully differentiable by applying a continuous approximation.

Another notable approach of applying VAEs to text was recently proposed in Guu et al. (2017), where generation is treated as a prototype-then-edit task – sample a prototype sentence from the training corpus and then edit it into a new sentence. Unlike conventional VAEs where the encoder packs the whole sentence into a latent vector, Guu et al. (2017) choose the latent vector to represent an edit that transforms an input prototype into a new sentence.

Finally, autoencoders as a key technique in unsupervised representation learning have been widely applied in NLP tasks to regularize language models and sequence-to-sequence models (Dai and Le, 2015), for supervised machine translation (Zhang et al., 2016), and more recently, for enabling unsupervised machine translation (Artetxe et al., 2017; Lample et al., 2017).

## 3 Background

In this section, we briefly review two previously proposed types of generative models which we adopt in this work: variational and adversarial autoencoders.

Both are autoencoders consisting of two components: an encoder $E$, which transforms an input $x$ to an embedding (latent code) $z$, and a decoder (generator) $G$, which produces a reconstruction of $x$ from $z$. A prior distribution $p(z)$ is imposed on the embedding space and the model is trained to match the aggregated posterior $p_E(z) = \int_x p_{\text{data}}(x)p_E(z|x)dx$ to the prior. The two models differ in the way they achieve this goal: a VAE includes a KL divergence term in its cost function, while AAEs employ an adversarial training objective.

### 3.1 Variational autoencoder

A *variational autoencoder* (Kingma and Welling, 2013) maximizes a lower bound on the marginal log-likelihood:

$$\log p_G(x) \geq \mathbb{E}_{p_E(z|x)}[\log p_G(x|z)] \\ -\text{KL}(p_E(z|x) \,||\, p(z)).$$

The first term is the log-probability of reconstructing the input $x$ given the latent vector $z$ sampled from the posterior distribution. The second term is the negative KL divergence from the prior to the posterior, which effectively acts as a regularizer, pushing the posterior closer to the prior.

A standard Gaussian is usually chosen as the prior distribution, and the posterior (the output distribution of the encoder) is modelled as a diagonal Gaussian to allow for gradient back-propagation using a reparameterization trick.

A VAE for text, as proposed by Bowman et al. (2016), uses an RNN encoder and decoder. The authors use KL cost annealing (gradually increasing the weight of the KL term from 0 to 1) and word dropout (randomly masking out tokens from the decoder's input during training) to encourage the decoder to make use of the latent vector produced by the encoder.

### 3.2 Adversarial autoencoder

*Adversarial autoencoders* (Makhzani et al., 2015) regularize the embedding space by means of adversarial training. The model is extended with an adversarial network (discriminator) $D$, which is trained to predict whether a given vector $z$ is a sample from the imposed prior distribution $p(z)$ or an embedding produced by the encoder:

$$\max_D \mathbb{E}_{p(z)}[\log p_D(z)] + \\ \mathbb{E}_{p_E(z|x)p_{\text{data}}(x)}[\log(1 - p_D(z))].$$

Here, $p_D(z)$ is the probability, predicted by the discriminator, that $z$ is a genuine sample from the prior distribution.

Meanwhile, the autoencoder is trained in two alternating SGD steps: In the *reconstruction phase*, we optimize the standard reconstruction objective:

$$\max_{E,G} \mathbb{E}_{p_E(z|x)p_{\text{data}}(x)}[\log p_G(x|z)].$$

In the *regularization phase*, the encoder is trained to fool the discriminator so that the latter is unable to distinguish the encoder outputs from the samples coming from $p(z)$:

$$\max_E \mathbb{E}_{p_E(z|x)p_{\text{data}}(x)}[\log p_D(z)].$$

Note that $p(z)$ can now be an arbitrary distribution, as long as we can sample from it.

## 4 Models

We will now describe the details of the models we examine in this work, our modifications to them, and our choice of prior distributions.

**VAE.** In the variational autoencoder, we adhere to the commonly used Gaussian prior and diagonal Gaussian posterior. We employ KL term annealing and word dropout as in Bowman et al. (2016). In one of the settings (VAE-BOW), we replace word dropout with the bag-of-words loss of Zhao et al. (2017b).

**AAE.** For adversarial autoencoders, we experiment with two kinds of prior distributions: a standard Gaussian and a uniform distribution on the unit sphere (in Euclidean space).

In the case of a Gaussian prior, we use two types of posterior distributions: a diagonal Gaussian parameterized by the encoder, i.e. $p(z|x) = \mathcal{N}(z;\, \mu_E(x),\, \sigma^2_E(x))$, and a deterministic posterior, where the encoder produces a single $z = E(x)$ for each input. We refer to the two resulting models as AAE-GAUSS and AAE-GAUSS-DET, respectively.

In the spherical case (AAE-SPH), we normalize the output of the encoder to ensure that the aggregated posterior distribution is supported on the unit sphere. During training, we add Gaussian noise to the normalized embeddings before passing them to the decoder and the discriminator. The variance of this noise is either fixed or exponentially decayed over time.

Unlike Makhzani et al. (2015), we combine the reconstruction and regularization phase in one training objective:

$$\max_{E,G} \mathbb{E}_{p_E(z|x)p_{\text{data}}(x)}[\log p_G(x|z) + \lambda \log p_D(z)].$$

We use $\lambda = 20$ except where stated otherwise.

To further regularize the decoder, we apply dropout with a keep probability of 0.4 on the LSTM inputs and states (Gal and Ghahramani, 2016).

**ARAE.** Adversarially regularized autoencoders (Zhao et al., 2017a) are similar to AAEs, but instead of imposing a prior distribution on the embeddings, they learn a flexible prior and employ adversarial training to match it to the aggregated posterior. We use the original ARAE implementation,[2] modified to perform decoding from the mean of the posterior distribution. We evaluate two ARAE configurations: the defaults used by Zhao et al. (2017a) and a modified setup with hyper-parameters and training time matching our models.

**Plain AE.** We also include a plain autoencoder, which isn't endowed with a means of controlling the aggregated posterior. However, in order to be able to draw samples from the model, we still assume a prior distribution on the embeddings – either Gaussian (AE-GAUSS-DET) or spherical (AE-SPH). These autoencoders are equivalent to their adversarial counterparts with $\lambda$ set to 0.

Note that while there is no explicit control over the embedding space of AE-GAUSS-DET, the outputs of the AE-SPH encoder are constrained to the unit sphere (although a uniform distribution is not enforced).

## 5 Experimental setup

We train and evaluate all models on a public corpus consisting of 200,000 sentence summaries extracted from news articles[3] (Filippova and Altun, 2013). We perform unsupervised sub-word tokenization using SentencePiece.[4]

Each of the models is evaluated on two different tasks:

- **Sampling.** As a pure unconditional generative model, drawing random samples from the prior distribution and performing greedy decoding on each of the samples. This allows us to measure how well the model approximates the underlying data distribution.

- **Reconstruction.** As an autoencoder, measuring the reconstruction quality. For this task, we first encode the input sentence, take the latent vector $z$ to be the mean of the posterior distribution, and then run greedy decoding.

[2] https://github.com/jakezhaojb/ARAE

[3] https://github.com/google-research-datasets/sentence-compression

[4] https://github.com/google/sentencepiece

We give examples of generated sentences in the supplementary material.

### 5.1 Sampling evaluation

To evaluate an unconditional generative model for text, we would like to make sure that (a) the generated sentences are correct with respect to the language used in the training data, and (b) the generated sentences reflect the diversity of expressions in the training data, i.e. the model avoids mode collapse. In order to capture both requirements, we use a number of different evaluation metrics.

**Cross entropy.** A natural way to evaluate a probabilistic model is cross entropy:

$$\mathbb{E}_{p_{\text{data}}(x)}[-\log p_G(x)]. \quad (1)$$

Note, however, that $p_G(x) = \mathbb{E}_{p(z)}[p(x|z)]$ is intractable for any given $x$. Following Zhao et al. (2017a), we approximate $p_G(x)$ using an RNN language model trained on 100,000 model samples; then, to obtain an estimate of (1), we evaluate this LM on the test set.[5]

We are also interested in 'reverse cross entropy', i.e. the expected negative log-probability of samples from the model with respect to the true data distribution:

$$\mathbb{E}_{p_G(x)}[-\log p_{\text{data}}(x)]. \quad (2)$$

This can be thought of as a measure of plausibility (fluency) of the generative model's outputs. Again, $p_{\text{data}}(x)$ is unknown, but can be approximated using a language model. Therefore, to estimate (2), we score the samples from each model using a pre-trained RNN LM. The model is trained on a large news corpus from the English Gigaword.[6]

**Fréchet distance.** In addition, motivated by a comprehensive study of various GAN models (Lucic et al., 2017) where the authors use Fréchet Inception Distance (Heusel et al., 2017) extensively and demonstrate that it is superior to the Inception Score (Salimans et al., 2016), we experiment with an equivalent metric for text – *Fréchet InferSent Distance* (FID) – to measure the distance between the generative distribution and the data distribution. FID measures the Wasserstein-2 distance (Vaserstein, 1969) between two Gaussians, whose means and covariances are taken from embeddings of the real and generated data (i.e. samples from $p_{\text{data}}$ and $p_G$), respectively. To our knowledge, this is the first time that this idea is applied to evaluating generative models for text. Different from the negative log-likelihood metrics discussed above, FID directly measures the distance between distributions, hence it offers an additional angle for comparing generative models whose goal is to learn to recover the true data distribution.

We compute FID between 10,000 sentences generated from the model and taken from the test set, respectively. To obtain their embeddings, we use a pre-trained general purpose sentence embedding model, InferSent (Conneau et al., 2017), which encodes each sentence as a 4,096-dimensional vector. We chose InferSent for computing the FID metric on sentence samples because it has been shown to provide state-of-the-art results on various sentence representation tasks, and is domain-independent to a large extent.

### 5.2 Reconstruction evaluation

The metrics in the previous section quantify the quality and diversity of samples generated while conditioning the decoder on a sample from the prior $p(z)$. Another way to gauge the diversity of sentences the model can represent is to measure how accurately it can reconstruct a given input. We express the reconstruction error as negative log-likelihood (NLL) and BLEU-3 and ROUGE-3 scores computed with the input sentence as a reference.

### 5.3 Human evaluation

We also evaluate the models based on subjective human judgment, focusing on the two tasks mentioned above: sampling and reconstruction.

For sampling, we decode sentences from random points in the embedding space and ask human raters to rate them on a 5-point Likert scale according to their fluency. Raters are trained so that a score of 1 corresponds to gibberish, 3 corresponds to understandable but ungrammatical sentences, and 5 corresponds to naturally constructed and understandable sentences.

For reconstruction, we present the raters with a sentence and its reconstruction produced by one of the models. Besides assessing fluency, the raters are asked to provide another score on a 5-point

[5] Note that Zhao et al. (2017a) use an equivalent metric, but refer to it as 'reverse perplexity'.

[6] https://catalog.ldc.upenn.edu/ldc2003t05

Likert scale measuring how well the output reflects the original meaning (this score is referred to as relevance in the following). A score of 1 corresponds to an unrelated sentence, 3 corresponds to a reasonably good paraphrase, and 5 corresponds to a perfect reconstruction, i.e. an identical output or a semantically equivalent paraphrase.

The evaluation was done using a crowdsourced rating platform. For both tasks, we evaluated a sample of 200 sentences from each model, employing three raters per item. The results were calculated as an average of the median sentence ratings. For 84% of the items, there was a majority score, i.e. at least two of the three raters chose the same of the 5 possible scores for the item.

## 6 Results

### 6.1 Quantitative evaluation

The results are shown in Table 1 (automatic evaluation) and Tables 2 and 3 (human evaluation). Results on samples from the training set and from an RNN LM are included for comparison. The RNN LM uses the same architecture and hyperparameters as the decoder of all the other models.

For the sampling task, one thing to notice is that there seems to be a trade-off between the quality and the diversity of the samples: models with a lower (i.e. better) reverse cross entropy and a higher fluency rating tend to have a higher (i.e. worse) forward cross entropy. In particular, the reverse cross entropy of some models (VAE and some -SPH models) is less than that of the real data – this is a clear sign that the model is suffering from a mode collapse. This is supported by the fact that these models also tend to have worse performance on reconstruction, which suggests that the set of sentences they are able to encode is less diverse.

Another important observation is that plain autoencoders with the spherical prior (AE-SPH) achieve relatively good results, on par with their adversarial counterparts (AAE-SPH). This suggests that the techniques applied in these models – constraining the embeddings to lie on a unit sphere and injecting noise – are sufficient for making the model learn to cover the sphere uniformly and be able to decode sentences from any given point on the sphere. The adversarial training seems to have little additional effect, if any at all.

In particular, AE-SPH with $\sigma = 0.1$ performs at least as well on sampling as all other types of models we evaluated:

- It achieves a superior forward cross entropy.
- Its FID is only slightly higher than for VAE ($d_{\text{w}} = 0.5$), which achieves the lowest (i.e. best) value.
- Although its reverse cross entropy is still below the real data threshold, it is higher than for VAEs, hence it arguably suffers less from the mode collapse problem.
- It achieves a higher fluency score than a LM and is only surpassed by the VAE.
- Finally, it outperforms VAEs on the reconstruction task by a large margin.

The effect of adversarial training on the Gaussian prior model (AAE-GAUSS-DET) seems to be more pronounced than in the spherical prior models – this is unsurprising as the non-adversarial variant (AE-GAUSS-DET) doesn't place any restrictions on the aggregated posterior, and therefore cannot be expected to be useful as a generative model. However, AAE-GAUSS-DET still has poor performance on sampling according to both automatic and human evaluation.

Regarding ARAE, it outperforms all other methods on almost all reconstruction metrics, but its results on sampling are rather poor, especially according to human ratings. This might be due to a more challenging dataset than in Zhao et al. (2017a), or simply because of the model's high sensitivity to hyperparameters, which is noted by the authors.

### 6.2 Embedding visualization

Fig. 1 shows t-SNE (van der Maaten and Hinton, 2008) projections in 2D of the encodings of ten random sentences from the test set. Each sentence has been encoded one hundred times with sampling from the posterior, then plotted with some additional noise in order to better visualize collapsed points.

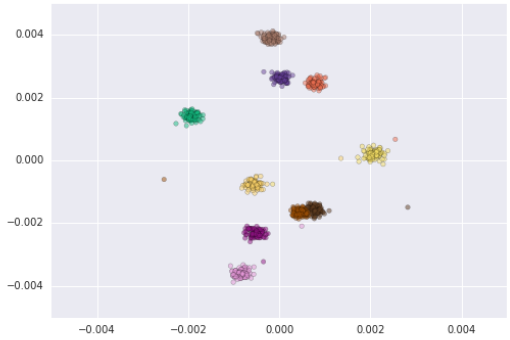
Plain AE-GAUSS-DET is deterministic, and each sentence is mapped to the same identical point all 100 times. This leads to a very high-quality reconstruction, but the embedding space is not smooth and sampling from random points in the prior would often produce unreadable outputs. Plain VAE exhibits the opposite behaviour: all 10 inputs are encoded into large, heavily overlapping

| Model | Sampling | | | Reconstruction | | |
|---|---|---|---|---|---|---|
| | **Forward** | **Reverse** | **FID** | **BLEU** | **ROUGE** | **NLL** |
| real data | 73.11 | 75.38 | 0.4193 | — | — | — |
| LM | 78.50 | 88.75 | 0.6267 | — | — | — |
| VAE ($d_{\text{w}} = 0.5$) | 79.75 | 65.46 | **0.6562** | 10.27 | 20.52 | 66.3 |
| AAE-SPH ($\sigma = 0.075$) | **74.01** | 82.34 | 0.6622 | 50.90 | 59.93 | 36.0 |
| AAE-SPH ($\sigma = 0.1$) | 76.28 | 66.73 | 0.6632 | 35.19 | 46.53 | 42.1 |
| AE-SPH ($\sigma = 0.1$) | 75.77 | 67.98 | 0.6635 | 37.56 | 49.42 | 26.3 |
| AAE-SPH ($\sigma = 0.05$, $\lambda = 10$) | 74.28 | 103.33 | 0.6749 | 60.39 | 68.14 | 31.3 |
| AE-SPH ($\sigma = 0.05$) | 74.69 | 101.27 | 0.7403 | 63.22 | 70.08 | 14.3 |
| ARAE | 80.48 | 94.51 | 0.7871 | **72.21** | 75.11 | **7.1** |
| AE-SPH ($\sigma \to 0$, $d = 1$) | 79.62 | 117.911 | 0.8748 | 11.75 | 20.80 | 105.8 |
| ARAE (default) | 99.67 | 73.37 | 0.8860 | 18.38 | 26.08 | 33.4 |
| VAE-BOW ($d_{\text{w}} = 1$) | 87.75 | 63.06 | 1.0150 | 2.03 | 11.43 | 88.9 |
| AAE-GAUSS-DET ($\lambda = 10$, $d = 1$) | 88.49 | 116.18 | 1.1433 | 68.01 | 73.29 | 23.6 |
| VAE ($d_{\text{w}} = 0.75$) | 112.16 | **59.59** | 1.2440 | 2.04 | 11.30 | 70.5 |
| AE-GAUSS-DET ($d = 1$) | 107.16 | 71.12 | 3.0839 | 71.14 | **76.25** | 9.9 |

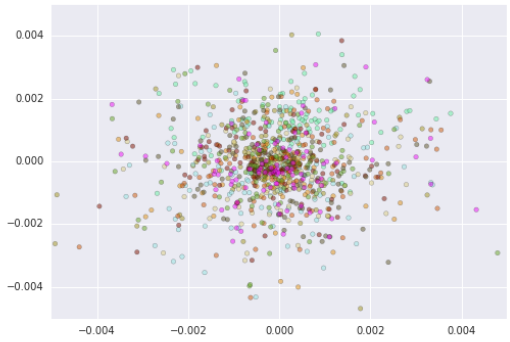
Table 1: Automatic evaluation results. **Forward**: 'forward cross entropy', i.e. the negative log-likelihood (NLL) of a LM trained on the samples from each model and evaluated on the test set; **Reverse**: 'reverse cross entropy', i.e. the NLL of a LM trained on real data and evaluated on the model samples; **FID**: Fréchet InferSent Distance. The reconstruction section reports the **NLL**, **BLEU** and **ROUGE** w.r.t. the input sentence. $\sigma$ denotes the standard deviation of the noise added to the sentence embeddings during training. $d$ and $d_{\text{w}}$ denote the RNN dropout and word dropout keep probability in the decoder, respectively ($d = 1$ means no dropout). 'Real data' corresponds to samples from the training set.

| Model | Relevance | Fluency |
|---|---|---|
| real data | — | 4.42 |
| AAE-GAUSS-DET ($\lambda = 10$, $d = 1$) | **3.54** | 3.71 |
| ARAE | 3.35 | 3.56 |
| AAE-SPH ($\sigma = 0.075$) | 2.76 | 3.53 |
| AE-SPH ($\sigma = 0.1$) | 2.54 | 3.53 |
| AAE-SPH ($\sigma = 0.1$) | 2.40 | 3.54 |
| AE-SPH ($\sigma \to 0$, $d = 1$) | 1.73 | 2.33 |
| ARAE (default) | 1.48 | 2.51 |
| VAE ($d_{\text{w}} = 0.5$) | 1.39 | **3.87** |

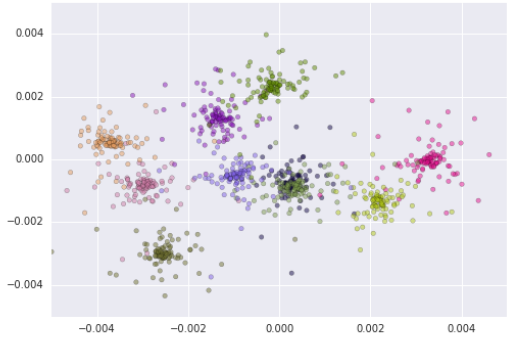
Table 2: Human evaluation results for the reconstruction task. Each score is on a scale of 1 to 5. The readability score for real data from Table 3 is included for comparison.



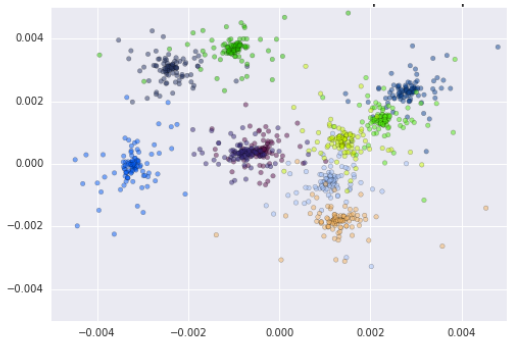

(a) AAE-GAUSS-DET




(b) VAE




(c) AE-SPH




(d) AAE-SPH

Figure 1: t-SNE visualization of 100 different encodings (samples from the posterior distribution) of 10 sentences, using various models.

| Model | Fluency |
|---|---|
| real data | 4.42 |
| VAE ($d_{\text{w}} = 0.5$) | **3.46** |
| AE-SPH ($\sigma = 0.1$) | 3.07 |
| AAE-SPH ($\sigma = 0.1$) | 2.83 |
| LM | 2.69 |
| AAE-SPH ($\sigma = 0.075$) | 2.61 |
| ARAE (default) | 2.08 |
| AE-SPH ($\sigma \to 0$, $d = 1$) | 1.85 |
| ARAE | 1.68 |
| AAE-GAUSS-DET ($\lambda = 10$, $d = 1$) | 1.53 |

Table 3: Human evaluation results for the sampling task. Each score is on a scale of 1 to 5.

regions of the embedding space. This hints at why this model performs poorly for reconstruction and has very good quality sampling from any random point in the embedding space.

Finally AAE-SPH and AE-SPH display similar behaviours, with sentences mapped into smooth regions in the space without significant overlap in the projections.

While not a quantitative study by itself, the plots are consistent with the observed results for sampling and reconstruction described above.

## 7 Conclusions

We introduced a rigorous evaluation scheme for generative models for text. In addition to previously proposed metrics, we proposed the Fréchet InferSent Distance, adopted from the field of image generation.

Three families of generative models (plain, variational and adversarially regularized autoencoders) have been thoroughly compared, under different regularization strategies. The qualitative evaluation shows that no model outperforms the others under all circumstances, with VAE being the strongest for sampling, but suffering from mode collapse and poor reconstruction performance. The rest of the models represent compromises between good sampling and reconstruction, and as we have demonstrated, the trade-off between these two can be controlled using simple regularization techniques.

## Acknowledgments

We thank Sylvain Gelly for useful feedback and pointing to FID metric.

## References


Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2017. Unsupervised neural machine translation. volume abs/1710.11041.

Olivier Bousquet, Sylvain Gelly, Ilya Tolstikhin, Carl-Johann Simon-Gabriel, and Bernhard Schoelkopf. 2017. From optimal transport to generative modeling: the VEGAN cookbook.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *CoNLL*.

Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Variational lossy autoencoder. volume abs/1611.02731.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364* .

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning. volume abs/1511.01432. http://arxiv.org/abs/1511.01432.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *EMNLP*.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., pages 1019–1027.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2017. Generating sentences by editing prototypes. *CoRR* abs/1709.08878.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs trained by a two time-scale update rule converge to a local Nash equilibrium.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *ICML*.

Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aäron van den Oord, Alex Graves, and Koray Kavukcuoglu. 2016. Neural machine translation in linear time. volume abs/1610.10099.

Diederik P. Kingma and Max Welling. 2013. Auto-encoding variational Bayes. *CoRR* abs/1312.6114.

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. volume abs/1711.00043. http://arxiv.org/abs/1711.00043.

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. 2017. Are GANs created equal? A large-scale study. volume abs/1711.10337. http://arxiv.org/abs/1711.10337.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow. 2015. Adversarial autoencoders. *CoRR* abs/1511.05644.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Cambridge, U.K.: Cambridge University Press.

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training GANs. volume abs/1606.03498. http://arxiv.org/abs/1606.03498.

Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. 2017. A hybrid convolutional variational autoencoder for text generation. In *EMNLP*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. In *Journal of Machine Learning Research*.

Leonid Nisonovich Vaserstein. 1969. Markov processes over denumerable products of spaces, describing large systems of automata. *Problemy Peredachi Informatsii* 5(3):64–72.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. In *ICML*.

Biao Zhang, Deyi Xiong, and Jinsong Su. 2016. Variational neural machine translation. volume abs/1605.07869.

Junbo Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, and Yann LeCun. 2017a. Adversarially regularized autoencoders for generating discrete structures. *CoRR* abs/1706.04223.

Tiancheng Zhao, Ran Zhao, and Maxine Eskénazi. 2017b. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *ACL*.