A Transformer-Based Variational Autoencoder for Sentence Generation

1st Danyang Liu, 2nd Gongshen Liu*
School of Electronic Information and Electrical Engineering
Shanghai Jiao Tong University
Shanghai, China
{danyliu, lgshen}@sjtu.edu.cn

***Abstract*—The variational autoencoder(VAE) has been proved to be a most efficient generative model, but its applications in natural language tasks have not been fully developed. A novel variational autoencoder for natural texts generation is presented in this paper. Compared to the previously introduced variational autoencoder for natural text where both the encoder and decoder are RNN-based, we propose a new transformer-based architecture and augment the decoder with an LSTM language model layer to fully exploit information of latent variables. We also propose some methods to deal with problems during training time, such as KL divergency collapsing and model degradation. In the experiment, we use random sampling and linear interpolation to test our model. Results show that the generated sentences by our approach are more meaningful and the semantics are more coherent in the latent space.**

***Index Terms*—variational autoencoder, text generation, self-attention, transformer**

I. Introduction

Generative models have a pivotal role in natural language understanding due to their use as language models in unsupervised learning with unlabeled data. Many natural language tasks can be considered as generative tasks with task-specific features, such as machine translation [1], abstractive summarization [2], dialogue modelling [3], etc.

Previous research has established that RNN-based generative models hold state of the art results in most text generation tasks [4]. These models generate sentences word-by-word, making it capable of modeling complex distributions over sequences and capturing long-term dependences. However, by decomposing a sentence prediction task into a series of next-word predictionsRNN model does not capture any interpretable representation of high-level features such as topic or syntactic properties.

In this paper, we focus on variational autoencoder(VAE) [5], a generative model which could explicitly capture high-level features in a continuous latent variable. The architecture of VAE is simple: Input sentence is mapped to a continuous latent space $P(X)$ like multivariate Gaussian distribution by the encoder, and then a latent representation is sampled from this continuous space $P(X)$. The objective of the decoder is to reconstruct the input sentence from the latent representation. At test time, when we want to generate new samples, we simply input values of $z \sim P(X)$ into the decoder.

* Gongshen Liu is corresponding author.

Although the VAE-based generative models have achieved good results when applied to images generation [7] and speech generation [8], their application to text generation has been far less successful. Studies of [9], [10] show that the major difficulty of training a VAE model is the collapse of the KL divergence(one of the latent loss) to zero. In this case, the representation from continuous latent space will be ignored and the model will degenerate into a standard language model. This is mainly due to the high modeling capabilities of the RNN-based decoders, which can achieve low reconstruction errors with little history, without relying on the latent vector generated from the encoder. In addition, although RNN has strong modeling abilities, it will consume more time cost because of the word-by-word architecture.

Our contributions are as follows: Firstly, we propose a novel VAE model for sentence generation. Contrary to previous research, where both encoder and decoder layers are LSTMs, our model is based on transformer structure. To the best of our knowledge, we are the first to apply transformer architecture in VAE models for text generation. To enhance our model's ability of generating long sequences, we added an LSTM layer at the end of the decoder. We also analyze some of the obstacles during training time and proposed our solutions. We evaluate our model on sentence generation and linear interpolation. Experimental results show that the choice of transformer architecture can achieve faster convergence and better performance than the previous LSTM-based VAE.

The remaining part of the paper proceeds as follows: In Section 2 we describe the proposed transformer-based VAE architecture and the optimization difficulties, as well as some background knowledge about VAE and transformer. Section 3 described the experimental setup and analysis the results of various evaluations. Related works are presented in Section 4. Finally, Section 5 concludes the paper and discusses future work.

II. Model

In this section, we begin by describing the background of VAE and transformer architecture. Then we introduce the proposed transformer-based VAE architecture. Finally, we discuss optimization difficulties and propose some solutions.

978-1-7281-2009-6/$31.00

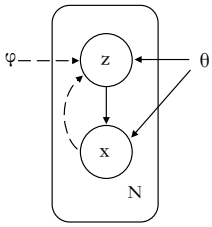

Fig. 1. Variational Auto-Encoder. The solid line indicates the generation model, and the dashed line indicates the variational approximation.

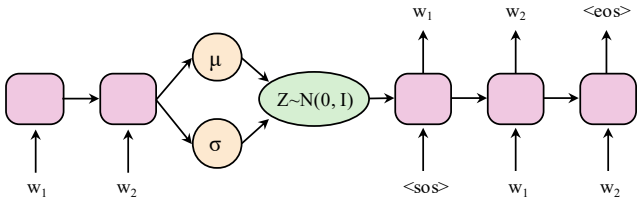

Fig. 2. LSTM-VAE model of [9]

A. Background on VAE

The variational autoencoder (VAE) [5] is a deep generation model with latent variable as shown in Fig. 1.

Suppose that the observed datapoint X is a random vector in a high-dimensional space, and the hidden variable Z is a relatively low-dimensional space. The joint probability density function of this generative model can be decomposed into:

$$p(x, z|\theta) = p(x|z, \theta)p(z|\theta) \tag{1}$$

$p(z|\theta)$ is the prior distribution of the hidden variable z. $p(x|z, \theta)$ is the conditional probability density function of the observed variable x when z is known, and θ indicates the parameter of the two density functions. In general, we can assume that $p(z|\theta)$ and $p(x|z, \theta)$ are some kind of parameterized distribution family, such as a normal distribution. The idea of VAE is to use neural networks to model these two complex probability density functions separately.

We use neural networks to generate the variational distribution $q(z|\phi)$, dubbed the inference network. Theoretically $q(z|\phi)$ can be independent of x. But since the goal of $q(z|\phi)$ is to approximate the posterior distribution $p(z|x, \theta)$, which is related to x, this variational density function is generally written as $q(z|x, \phi)$. The input to the inference network is x and the output is a distribution $q(z|x, \theta)$. More specifically, the outputs are the mean and variance of a Gaussian distribution.

Similarly, a neural network is used to generate a probability distribution $p(x|z, \theta)$, dubbed the generation network. The input to the generated network is z and the output is the probability distribution $p(x|z, \theta)$.

It has been proved in [5] that the objective function of VAE is the evidence lower bound(ELBO), which is composed of the reconstruction term and the KL divergency:

$$\begin{aligned} J_{vae}(\theta, \phi; x) =& D_{KL}(q(z|x, \phi)||p(z)) \\ & - E_{z\sim q(z|x,\phi)}[log\, p(x|z, \theta)] \end{aligned} \tag{2}$$

Where $p(z)$ is a Gaussian distribution with zero mean and unit variance $N(0, I)$. θ and ϕ represent parameters of decoder and encoder respectively. D_{KL} is the Kullback-Leibler diveregence.

The second term in the objective function could be considered as the reconstruction error, which is the same as the normal autoencoder. KL divergence term could force the distribution of the latent variable z to be close to $N(0, I)$. At test time, we simply input values of $z \sim N(0, I)$ into the decoder and it will generate new samples.

Reference [9] proposed the first VAE model for text generation where both encoder and decoder are LSTM networks (Fig.2). The abbreviation LSTM-VAE will be used to refer to this model throughout this paper. The authors show that applying VAE to text generation is a huge challenge because the decoder is prone to ignore latent vectors and degenerate into an LSTM language model. Some tricks can be used to alleviate this problem, and we will discuss solutions in Section II-D. Next, we describe the transformer architecture, which is the core element in the proposed VAE model.

B. Self-attention Mechanism

Transformer is a new sequence-to-sequence framework proposed in [11], which relies entirely on self-attention mechanism without using any convolution or sequence-aligned RNNs. It shows remarkable success over existing algorithms for machine translation and other language understanding tasks.

Attention mechanism is the core of transformer model. Generally speaking, an attention function maps a query and a set of key-value pairs to an output. The output is a weighted sum of key values, where the weight is computed by a similarity function of the query with the corresponding key.

$$Attention(Q, K, V) = Correlation(Q, K) * V \tag{3}$$

In transformer the correlation function is implemented by a scaled dot-product attention as shown in Fig.3, which divide each dot products of query-key pairs by $\sqrt{d_k}$.

In addition, the transformer also uses a multi-head attention mechanism as shown in Fig.3. Firstly, the queries, keys, and values are projected to different subspaces h times using different linear projections. Then the self-attention function is calculated in each subspace, and finally these outputs are concatenated and projected again to obtain the final output.

$$\begin{aligned} MultiHead(Q, K, V) &= Concat(head_1, ..., head_h)W^o \\ where head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{4}$$

Where the matrices $W_i^Q \in \mathbb{R}^{d_{model}\times d_k}, W_i^K \in \mathbb{R}^{d_{model}\times d_k}, W_i^V \in \mathbb{R}^{d_{model}\times d_v}$ and $W^O \in \mathbb{R}^{hd_v\times d_{model}}$ represent the correspoding linear projections.

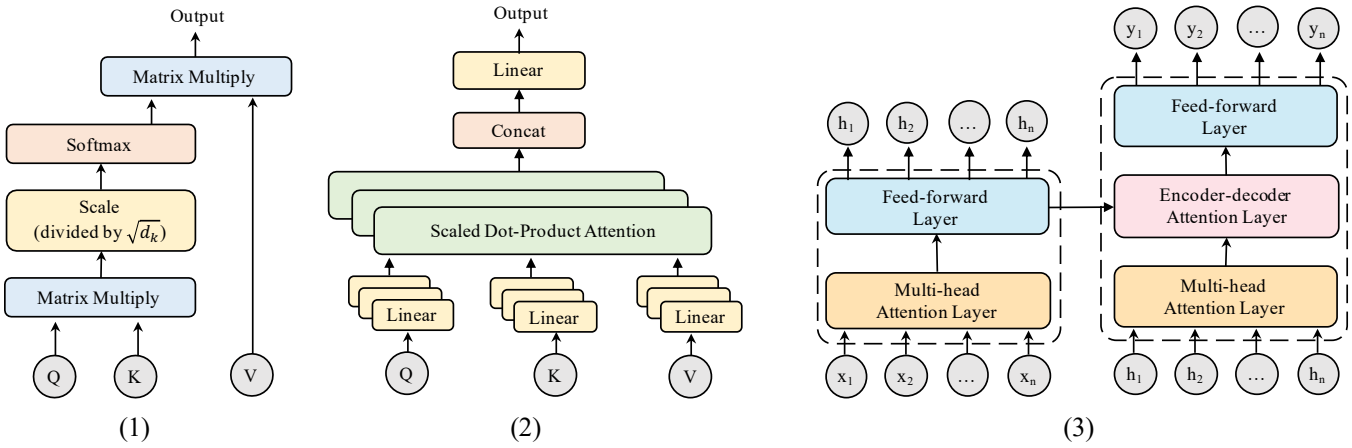

Fig. 3. Illustrations of Scaled Dot-Product Attention (1), Multi-Head Attention (2), and the transformer model (3) adapted from [11]. In panel (3), inside the left dashed line is the encoder, and the right is the decoder.

Transformer has an encoder-decoder structure as shown in Fig.3. The encoder encodes the input sentence $X = (x_0, x_1, x_2..., x_n)$ to a continous representations $H = (h_0, h_1, h_2..., h_n)$. The decoder then maps H to an output sequence $Y = (y_0, y_1, y_2, ..., y_m)$ in an auto-regressive manner.

Concretely speaking, an encoder layer is composed of two sub-layers, a multi-head self-attention layer and a simple position-wise fully connected feed-forward layer. In the multi-head self-attention layer of the encoder, keys, values, and queries come from the same place, more specifically, the outputs of the previous layer.

A decoder is composed of three different sub-layers. Besides the two sub-layer in encoder layer, the decoder inserts a multi-head attention layer over the output of the encoder stack, where the queries are the outputs of the previous decoder layer, and the keys and values are the outputs of the encoder.

There are two primary motivations for choosing transformer architecture instead of the recurrent ones: Firstly, transformer uses parallel computing to increase training speed. Contrary to the step-by-step prediction of RNN, all input elements in the transformer are equal and can be operated in parallel. Secondly, Long-term dependencies could be directly captured in transformer. As we know, it is difficult for RNN to learn the long-term dependencies because of its autoregressive features. In Transformer, due to the existence of self-attention, there is a direct interaction between any inputs, establishing a direct dependency no matter how far away the inputs are.

C. Transformer-based VAE

We apply transformer architecture to both encoder and decoder modules of VAE as shown in Fig.4. The encoder is a stack of standard transformer modules with the same architecture in [11]. But in the decoder, we remove the second sub-layer of standard transformer decoder. Because we want this deep generative model to generate samples from a random variable, which means in test time the encoder is removed and

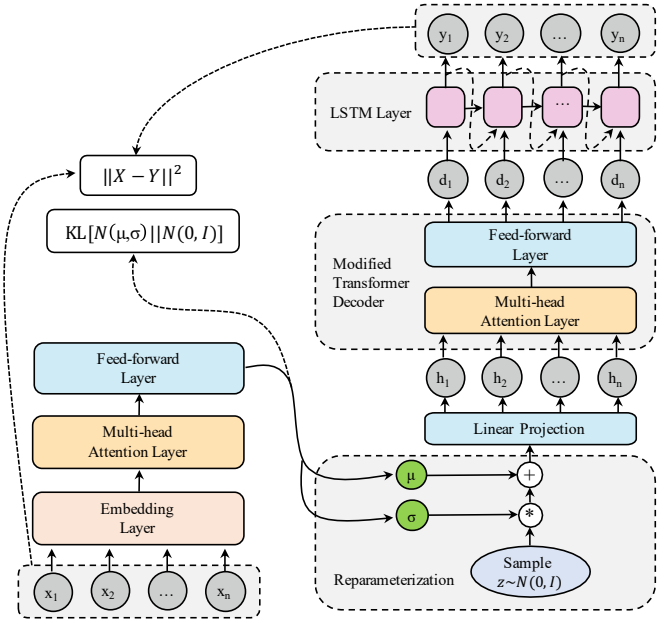

Fig. 4. An illustration of the self-attention-based VAE architecture.

we have no encoder outputs to calculate the encoder-decoder multi-head attention.

In order to compensate for the loss of encoder-decoder attention, we augment our decoder with an LSTM language model layer. This recurrent layer allows our model to focus more high-level semantic and stylistic features.

The training process of the model can be divided into 4 steps.

- Firstly, the encoder takes the embeddings of input sequences $X(x_1, x_2, ..., x_n)$once at a time and converts them into an mean μ and variance σ.
- Secondly, a latent vector z is sampled from a Gaussian distribution with some mean μ and covariance σ. Note that sampling operations are non-continuous and non-

differentiable, which means back-propagation doesn't work. The solution, called "reparameterization trick" in [5], is to move the sampling operation to an input layer, as shown in bottom right of Fig.4. Given the mean μ and covariance σ, we can sample from $N(\mu, \sigma)$ by first sampling $\beta \sim N(0, I)$, then computing $z = \mu + \sqrt{\sigma}\beta$.

- Thirdly, the latent vector z is projected to a input space of decoder, generating $(h_1, h_2, ..., h_n)$. The modified transformer decoder generate output $(d_1, d_2, ..., d_n)$ from $(h_1, h_2, ..., h_n)$. Then $(d_1, d_2, ..., d_n)$ are fed to LSTM language model and the output of LSTM layer is the reconstruction result.
- As described in (2), reconstruction error and KL divergency will be calculated. Then parameters are updated through back-propagation algorithm.

At test time, when we want to generate new samples, we simply input values of $z \sim N(0, I)$ into the decoder, which means the encoder is removed and only the decoder (the right part in Fig.4) remain working.

D. Optimization Method

Adding the LSTM language model can improve performance, but it also brings some training difficulties that are similar to what [6] have proposed.

When only the loss function of (2) is used, the KL divergency will quickly converge to a particularly small number around 0, which means that the model will degenerate into the traditional language model. In [6], [9] some researchers proposed a KL annealing algorithm, we found this method also helps our model to deal with zero KL term. Besides, as described in [12]–[14], we introduce random noise in the input sentence by making random input dropout.

KL term annealing means that we assign a weighting factor W to the KL divergency, which increases as the training steps increases. This allows the model to focus more on the reconstruction tasks in the early stages of training, while KL term can be slowly decreased. In this work, KL weight is defined as:

$$KL_{weight} = \frac{1}{1 + e^{-kn+b}} \tag{5}$$

Where k and b are parameters that control the rate of weight change, n is the training step. The total loss is then defined as:

$$\begin{aligned} J_{vae}(\theta, \phi; x) =& KL_{weight} * D_{KL}(q(z|x, \phi)||p(z)) \\ & - E_{z\sim q(z|x,\phi)}[log\, p(x|z, \theta)] \end{aligned} \tag{6}$$

Input dropout acts like a denoising autoencoder [15]. In order to avoid degeneration of the generative model and make the decoder truly learn the compositionality of its input. We want to limit the expressiveness of the language model, making the decoder rely more on the latent vector z.

III. Experiments

A. Setting

The standard Penn Tree-bank dataset [16] is used in our experiments. Vocabulary is build using words that appear at least 3 times. The dimension of word-embedding vectors is set to 256.

The encoder is a composed of a stack of 6 identical transformer layers with hidden dimension $h = 256$. In decoder we use a stack of 6 modified transformer decoder (as shown in middle right of Fig.4) with hidden dimension $h = 256$. The dimension of latent variable z is set to 16.

The model is trained using Adam optimization algorithmm [17] with learning rate $\alpha = 0.001$, momentum parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$ respectively, and $\varepsilon = 10^8$. We insert a batch normalization [18] layers in the middle of every two adjacent transformer layers. We use KL annealing in loss function as described in (6) and input random dropout algrithm with a probability of 0.2. The parameters k and b described in (5) are set to 0.0025 and 6.25 respectively.

B. Comparison with LSTM-VAE

We compare our model with the baseline method of LSTM-VAE. The effects of our proposed KL-annealing algorithm is shown in Fig.5. As shown in the right panel of Fig.5, the red line indicates the KL Loss without annealing algorithm. The KL term is kept at a small value near zero during the training time, causing the model to degenerate into a conventional language model. In this situation, the information of latent vector z is not fully utilized. When KL annealing algorithm is applied, as the blue line depicts, the KL term is limited in a reasonable range during training, making our model rely more on the latent vector z instead of being a conventional language model.

Fig.6 shows the total loss curves during training time when using LSTM-VAE and our Transformer-based VAE. Obviously, our model can converge faster and stay at a lower loss value.

As described in II-A, the ELBO (Evidence Lower Bound) indicates the total loss of VAE model. In Table.I, we present the ELBO values, reconstruction losses and KL divergency values of different models on the same test dataset. While the ELBO values are comparable, more attentions are paid into the latent vector z in Transformer-based VAE. This is reflected in higher KL Divergency value and the random samples from both models, presented in Table.II. The low reconstruction error indicates that our model also has better performance in the reconstruction task.

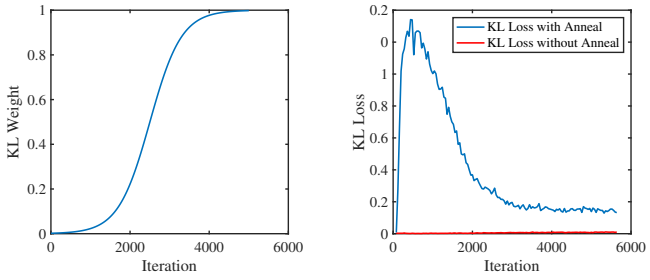

Fig. 5. Left: KL weight as described in (5). Right: Comparison of KL Loss when using or not using this KL annealing weight.

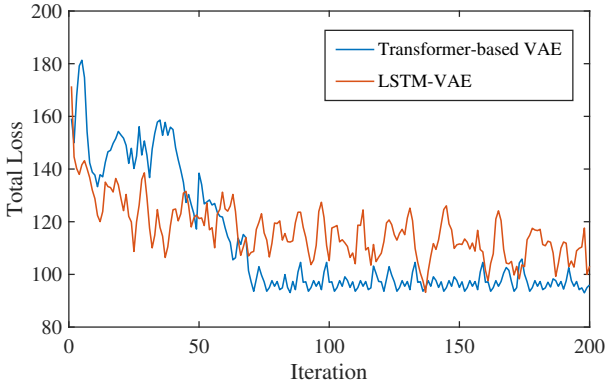

Fig. 6. Total loss values during training time of different model.

TABLE I
ELBO, KL AND RECONSTRUCTION TERMS

Model	ELBO	Recons.[a]	KL Term[b]
LSTM-VAE	105.75	97.23	7.41
Transformer-baed VAE	**99.31**	**81.66**	**17.65**

[a]Reconstruction Loss Value.
[b]KL Divergency Value.

C. Sentence Generation

We remove the encoder and feed randomly sampled vectors $z \sim N(0, I)$ to the decoder to test the generative ability of LSTM-VAE and our model. The results are shown in Table.II. LSTM-VAE is prone to generate many unknown symbols and repeating words, while the sentences generated by Transformer-based VAE are obviously more coherent and meaningful.

D. Linear operations in the latent space

The core idea of VAE is to map high-level semantic features into a continuous high-dimensional space, so we use linear interpolation to observe the semantic distribution between two points in high-dimensional space. The results are shown in Table.III. As Table.I shows, our model has a higher KL term, which suggests our model is better at taking advantage of the latent vector z and can interpolate more smoothly between sentences. Sentences representing begin and end points are bolded.

IV. Related Work

There are currently two of the most popular deep generative models, the Variational Autoencoder [5] and the Generative Adversarial Networks [19]. GAN has achieved breakthrough results in the field of image generation [20], [21]. But most NLP tasks are discrete, which means conventional GAN can not work. There is some work combined with reinforcement learning (RL) to solve this problem [22], but this is obviously not as efficient as VAE.

Reference [3] is the first work that applies VAE architecture in text generation task. The researchers have proved that it is possible to learn an LSTM-based text generative model from a latent variable. They also propose some evaluation methods similar to computer vision field: sample sentences by feeding the decoder randomly generated latent vectors z, and linearly interpolate between two sentence points in the latent space.

Recently, more attempts have been made to apply the VAE framework in sequence-to-sequence tasks. Reference [7] presents a hybrid model with CNN encoder and DCNN (deconvolution neural network). Reference [23] applies a special VAE with LSTM encoder and dilated CNN decoder to language model. Reference [25] apllies VAE to bag-of-words representations of paragraph and the answer selection problem achieving good results on both tasks. VAE is also used in some interesting areas, such as Chinese poetry generation and music generation. Reference [24] applied VAE in music generation for the first time. Reference [26] presents a thematic Chinese poetry generation model with a hybrid DCNN-RNN decoder.

Since the transformer model is first proposed in [11], it has been widely used in various sequence-to-sequence tasks and shows remarkable success over existing algorithms. Reference [27] use extractive summarization to coarsely identify salient information and a neural abstractive model based on transformer to generate the article. In [28], the authors generalize the transformer model to a sequence modeling formulation of image generation with a tractable likelihood. But there is no precedent for using transformers in VAE.

V. Conclusion

We have proposed a novel natural texts generative model based on VAE architecture, which is composed of a transformer-based encoder and a transformer-based decoder augmented with an LSTM language model layer. We also introduce a new KL-annealing function to alleviate the KL term collapsing.

In experiment sectionIII, we use LSTM-VAE as the baseline to compare the performance of the two different models in random sampling and linear interpolating.

Experimental results show that self-attention based architecture can solve the problem of KL-term vanishing and achieve better performance than the previous baselines. Meanwhile, the sentences generated by our model are more meaningful, and the sentences of linear interpolation in the continuous latent spaces are more coherent.

In the future, we would like to delve deeper into this direction and develop our model to a paragraph level text generation. Future work also includes the thematic text generation where a conditional variational autoencoder will be used [29].

Acknowledgment

This work was supported by the National Natural Science Foundation of China (61772337 and U1736207).

TABLE II
RANDOM SAMPLES

Model	Samples
LSTM-VAE	The @unk of the @unk @unk in the first time they have a @unk of @unk @unk or @unk @unk.
	The company ' s @unk @unk and @unk @unk.
	In the past two years we're entering the @unk of the movie.
	The pilots's action against the aba.
	The issue was quoted at @num @num.
Transformer-based VAE	The company also said it is considering a bid for the first time since @num.
	In the nine months the company earned @num million from @unk.
	I' m not going to see the @unk of my favorite business.
	The researchers said it would n't be reached to comment on the @unk.
	President of the exchequer nigel lawson said the @unk is scheduled to be @unk.

TABLE III
LINEAR INTERPOLATE

Model	Interpolate
LSTM-VAE	**@unk is a partner in @unk company.**
	@unk is a partner in @unk mass @unk of @unk .
	In addition the new york stock exchange by the @unk @unk .
	In addition the new york stock exchange trading @unk @unk n totaled n million shares .
	In new york stock exchange composite trading yesterday.
Transformer-based VAE	**The company said it will sell its @num stake in the first half of @num.**
	The company also said it would be able to sell its @unk business .
	The company's @unk system which includes the @num million in pretax profit margins .
	In the past two years the @unk of the company sell @unk the bank @unk .
	In the past two years the banks aren't permitted by the federal reserve bank.

REFERENCES

[1] D. Bahdanau, K. Cho, Y. Bengi, "Neural machine translation by jointly learning to align and translate," in arXiv preprint arXiv:1409.0473, 2014.

[2] A. M. Rush, S. Chopra, J. Weston, "A neural attention model for abstractive sentence summarization," in arXiv preprint arXiv:1509.00685, 2015.

[3] I. Vlad Serban, A. Sordoni, R. Lowe, L. Charlin, "A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues," in AAAI. 2017, pp. 3295-3301.

[4] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, Y. Wu, "Exploring the limits of language modeling," in arXiv preprint arXiv:1602.02410, 2016.

[5] D. P Kingma, M. Welling," Auto-encoding variational bayes," in arXiv preprint arXiv:1312.6114, 2013.

[6] S. Semeniuta, A. Severyn, E. Barth, "A hybrid convolutional variational autoencoder for text generation," in arXiv preprint arXiv:1702.02390, 2017.

[7] I. Gulrajani, K. Kumar, F. Ahmed, A. Ali Taiga, F. Visin, D. Vazquez, A. Courville, "Pixelvae: A latent variable model for natural images," in arXiv preprint arXiv:1611.05013, 2016.

[8] M. Fraccaro, S. Kaae Snderby, U. Paquet, O. Winther, "Sequential neural models with stochastic layers," in Advances in neural information processing systems. 2016, pp. 2199-2207.

[9] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, S. Bengio, "Generating sentences from a continuous space," in arXiv preprint arXiv:1511.06349, 2015.

[10] S. Semeniuta, A. Severyn, E. Barth, "A hybrid convolutional variational autoencoder for text generation," in arXiv preprint arXiv:1702.02390, 2017.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, . Kaiser, I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems. 2017, pp. 5998-6008.

[12] M. Artetxe, G. Labaka, E.Agirre, K. Cho, "Unsupervised neural machine translation," in arXiv preprint arXiv:1710.11041, 2017.

[13] A. M. Dai, Q. V. Le, "Semi-supervised sequence learning," in Advances in neural information processing systems. 2015, pp. 3079-3087.

[14] F. Hill, K. Cho, A. Korhonen, "Learning distributed representations of sentences from unlabelled data," in arXiv preprint arXiv:1602.03483, 2016.

[15] P. Vincent, H. Larochelle, Y. Bengio, P. Manzagol, "Extracting and composing robust features with denoising autoencoders," in Proceedings of the 25th international conference on Machine learning. ACM, 2008, pp. 1096-1103.

[16] M. P. Marcus, M. Ann Marcinkiewicz, B. Santorini, "Building a large annotated corpus of English: The Penn Treebank," in Computational linguistics, 1993, 19(2), pp. 313-330.

[17] D. P. Kingma, J. Ba, "Adam: A method for stochastic optimization," in arXiv preprint arXiv:1412.6980, 2014.

[18] S. Ioffe, C. Szegedy, ”Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in arXiv preprint arXiv:1502.03167, 2015.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, ”Generative adversarial nets,” in Advances in neural information processing systems. 2014, pp. 2672-2680.

[20] Denton E L, Chintala S, Fergus R. Deep generative image models using a laplacian pyramid of adversarial networks[C]//Advances in neural information processing systems. 2015: 1486-1494.

[21] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, ”Improved techniques for training gans,” in Advances in Neural Information Processing Systems. 2016, pp. 2234-2242.

[22] L. Yu, W. Zhang, J. Wang, Y. Yu, ”SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient,” in AAAI. 2017, pp. 2852-2858.

[23] Z. Yang, Z. Hu, R. Salakhutdinov, T. Berg-Kirkpatrick, ”Improved variational autoencoders for text modeling using dilated convolutions,” in arXiv preprint arXiv:1702.08139, 2017.

[24] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, D. Eck, ”A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music,” in arXiv preprint arXiv:1803.05428, 2018.

[25] Y. Miao, L. Yu, P. Blunsom, ”Neural variational inference for text processing,” in International Conference on Machine Learning. 2016, pp. 1727-1736.

[26] X. Yang, X. Lin, S. Suo, M. Li, ”Generating Thematic Chinese Poetry using Conditional Variational Autoencoders with Hybrid Decoders,” in Proceedings of the 27th International Joint Conference on Artificial Intelligence.

[27] P. J. Liu, M. Saleh, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, N. Shazeer, ”Generating wikipedia by summarizing long sequences,” in arXiv preprint arXiv:1801.10198, 2018.

[28] N. Parmar, A. Vaswani, J. Uszkoreit, . Kaiser, N. Shazeer, A. Ku, D. Tran, ”Image Transformer,” in arXiv preprint arXiv:1802.05751, 2018.

[29] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, M. Welling, ”emi-supervised Learning with Deep Generative Models,” in Advances in neural information processing systems. 2014, pp. 3581-3589.