

★ Member-only story

Evolution of Auto-encoders



Nikhil Verma · Follow

7 min read · May 7, 2022

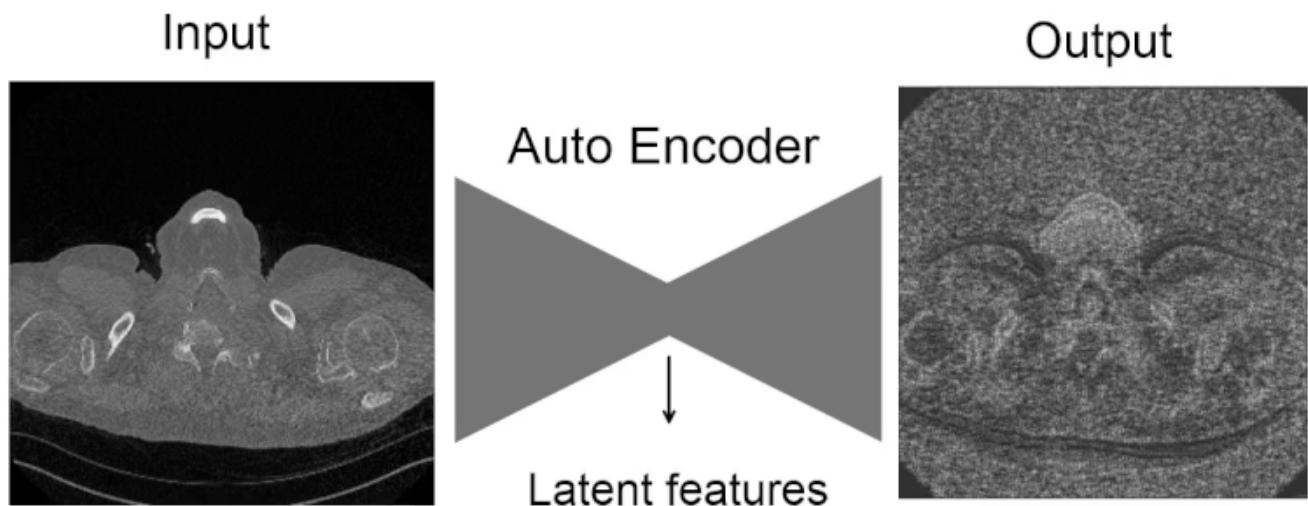


Listen



Share

... More



With an abundance of data in today's world, it really becomes important to not only store this data in compressed format but have some mechanism to transmit and compute as well in shortest dimensions possible. This could result in significant saving of cost in many formats. Where Storage and Retrieval is a concern for data engineers and transmission of data for network experts, computation is a field which everyone else is most interested in.

Since we know that some modalities(such as image, audio, video and text) of data are too large to be processed, it becomes need of the hour to find some representation of this data that is more syntactically crisp(has shorter dimension), semantically meaningful and could convey the same amount of information as the original data has. This is one of the prominent research domain that AI Researchers

have been focusing on since state-of-the-art deep learning models work with various modalities.

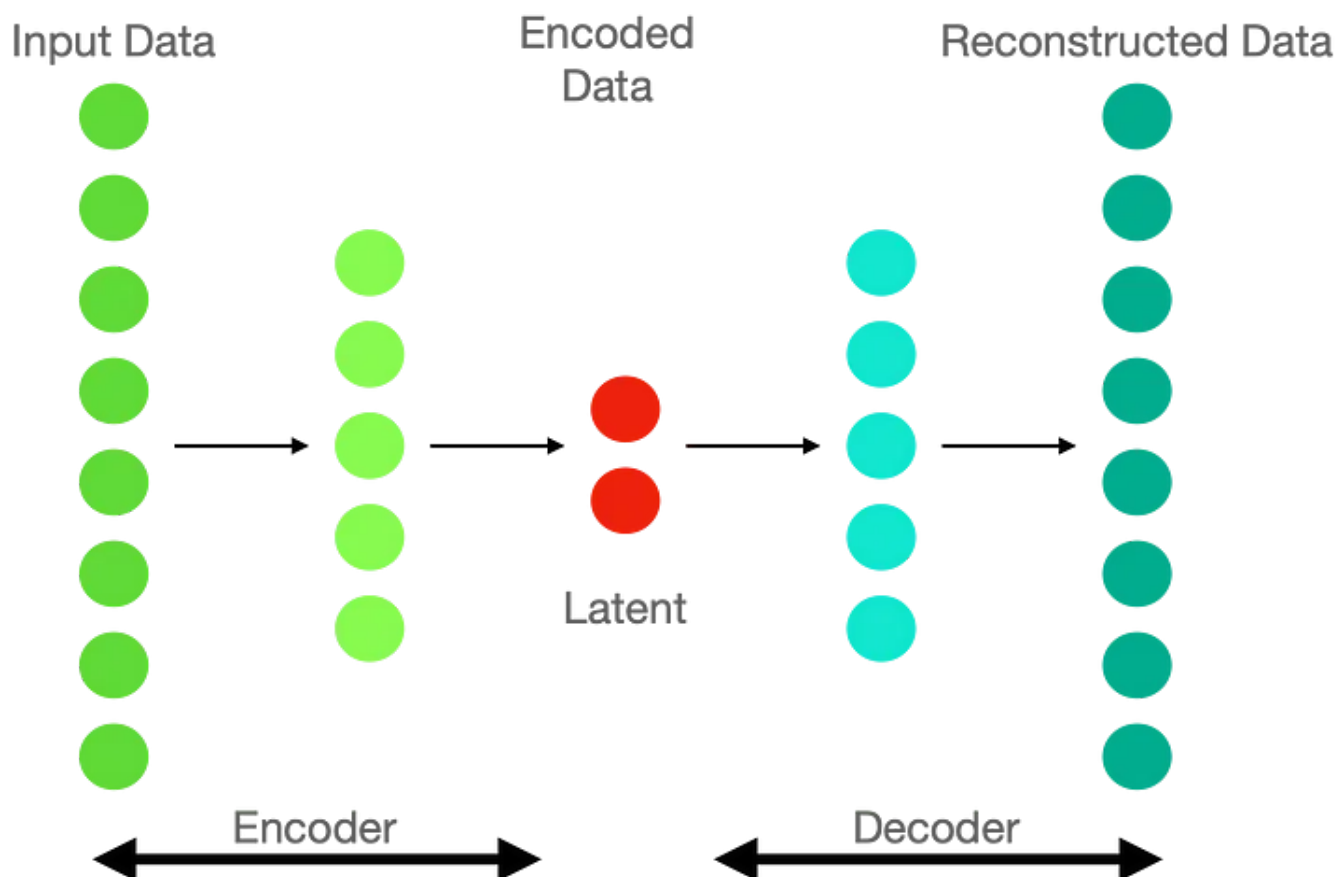
The discussion so far boils down to finding some latent space for the original data in hand. “Latent” comes from a latin word which means “to lie hidden”. Also called as hidden space is a compressed dimensional space beneath the original data(which exist in probably higher dimension), but focuses on important and semantically meaningful features of data in its representation.

Auto-encoders refer to a machine that could find latent space which are complicated non-linear functions of the raw data. With time and research efforts, Auto-encoders(AE) have grown from vanilla form to Variational auto-encoders(VAE), vector-quantised variational auto encoders(VQ-VAE) and discrete variational auto encoders(dVAE). The goal of article is to discuss about each of these architectures.

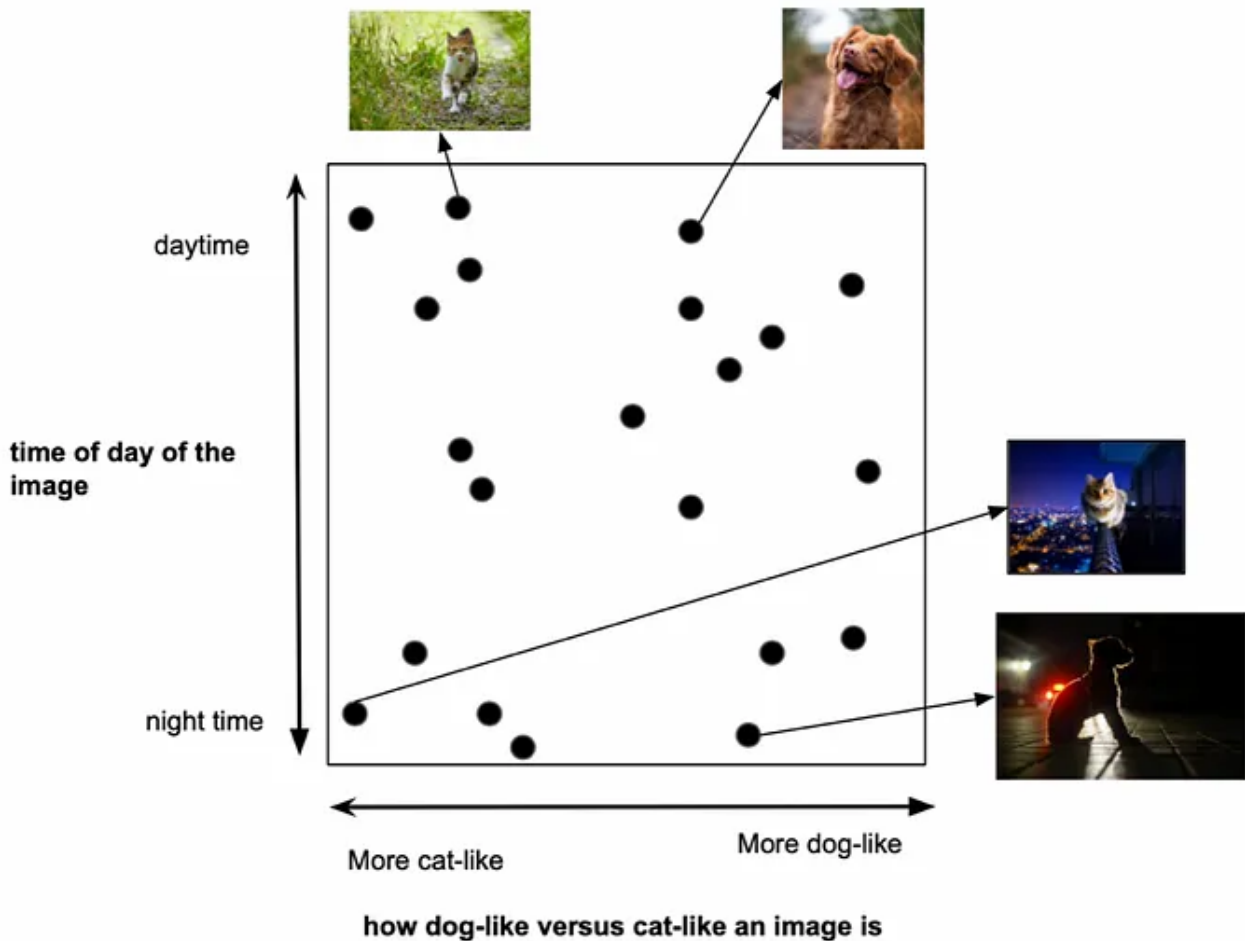
Since a simple RGB image itself has enough amount of data in it, we will focus on compressing image data in most of examples but methods are applicable on any other modality of data as well.

Auto-encoders

Auto-encoder is a deep learning model, which encodes the input data it receives in a smaller and more meaningful dimension. The layers in a neural network could be categorised as input layers, hidden layers and output layer. Using enough complicated non-linear activation functions we could learn complicated interactions of input features and at same time manage the size of this hidden representation. Auto-encoders further classify hidden layers into two parts — encoding layers and decoding layers. Encoding layers focus on reducing the size of data while decoding layers do the reverse. The architecture of AE looks as follows:-



Important to note over here is that latent encoded data has smaller dimension than the input. If I consider pixels of images of Cats and Dogs encoded to only 2-D variables, it would look like:-

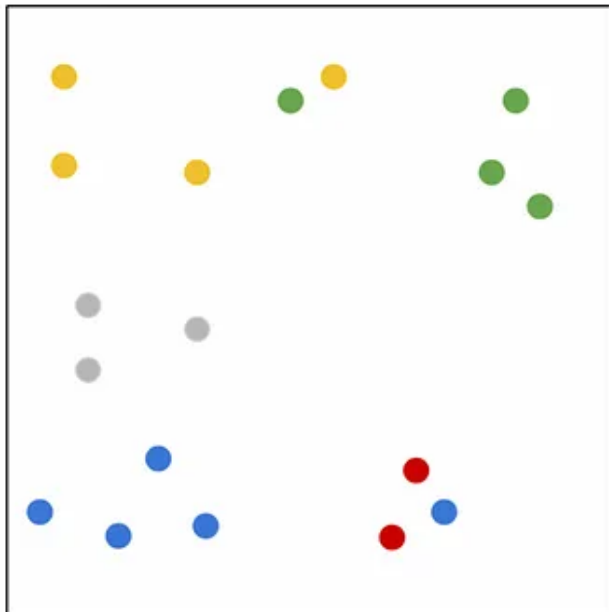


One could observe that in 2-D cats and dogs are randomly spread in R^2 space. This is the downside of a vanilla AE. Because ideally we would want our latent space to lump **semantically similar data points** next to each other and to place **semantically dissimilar points** far apart.

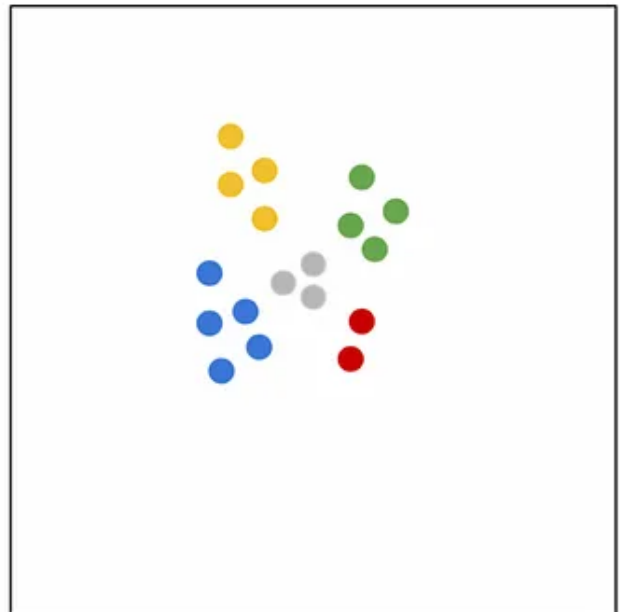
Variational Auto-encoders

VAEs are AE models that put a structure on the latent space. Model is trained in such a fashion that it explicitly assumes a prior distribution of the latent variable such as a continuous gaussian distribution. As a result the Latent space for VAE looks comparatively managed to vanilla AE.

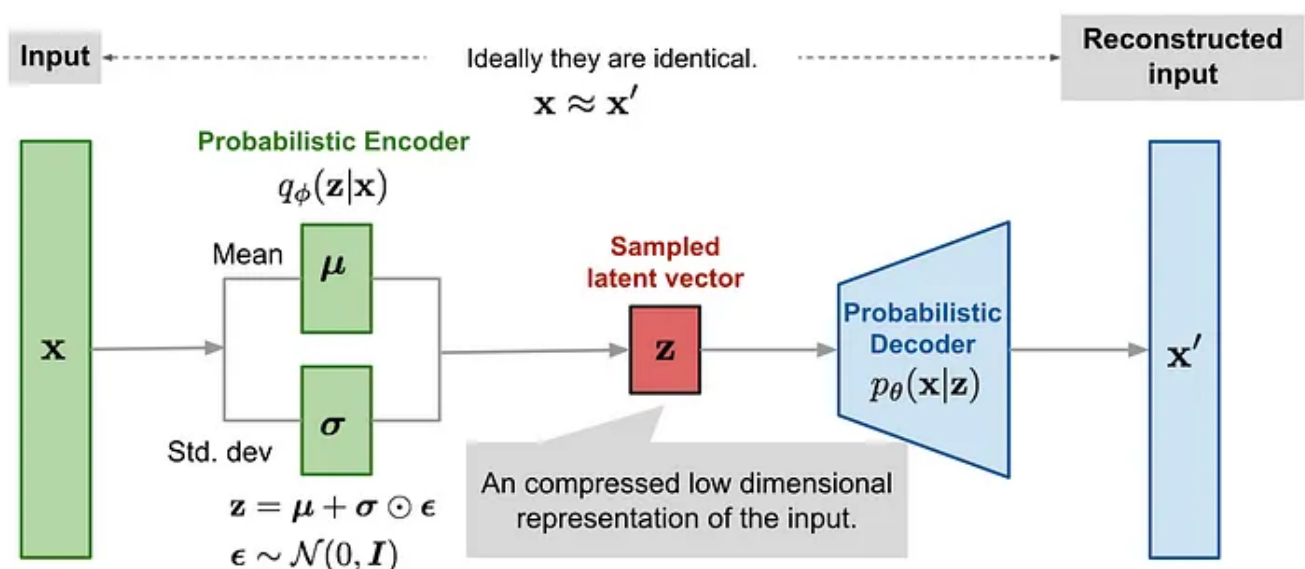
Messy Autoencoder Latent Space



Well Distributed VAE Latent Space



VAE overcome problems of AE by enforcing a probabilistic prior on the latent space and following is the architecture:-



Here the encoder module is responsible for learning a conditional distribution $q(\mathbf{z}|\mathbf{x})$ where \mathbf{z} is latent and \mathbf{x} is input. For a gaussian prior where $\mathbf{z} \sim p(\mathbf{z})$ (i.e. \mathbf{z} is drawn from a probability distribution which is gaussian), it outputs two parameters, mean and variance. Decoder module learns another neural network which maps latent space back to original space, i.e. $p(\mathbf{x}|\mathbf{z})$. The method used learn this newtork is maximise the ELBO(Evidence Lower Bound) which is derived from KL divergence. To back-propagate the derivative it uses re-parametrisation trick .

The cost function equation for VAE is given as:-

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

Here first term is a reconstruction loss, while second term tries to minimise the distance between learned conditional distribution and prior over the latent.

Vector-Quantised Variational Auto-encoder

Did you noted that VAE explicitly makes an assumption that latent variable is sampled from a Gaussian Distribution which is continuous in nature. But do a probability distribution $p(z)$ always need to be continuous and what would happen if its discrete. Well, one could argue that if $p(z)$ is discrete then we could only generate limited number of latent variables from that distribution. But, if distribution is quantised sufficiently large enough, then number of possible sample generation is a big number.

The fundamental difference between a VAE and a VQ-VAE is that:-

- VAE learns a continuous latent representation
- VQ-VAE learns a discrete latent representation

In general, a lot of the data we encounter in the real world favours a discrete representation. For illustration, discrete variables in image could be

Encoder



image to
discrete codes



56	73	67	23	81	19	...
----	----	----	----	----	----	-----

Decoder

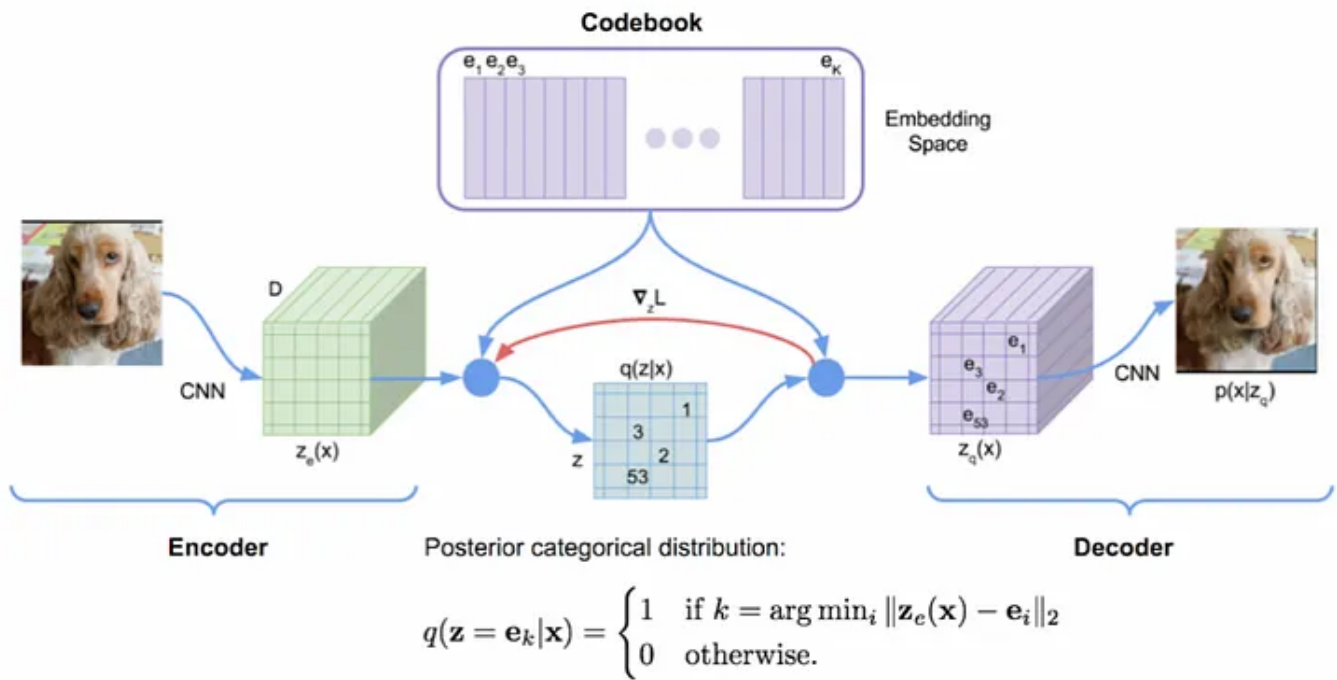
56	73	67	23	81	19	...
----	----	----	----	----	----	-----

discrete codes
to image



- type of object
- one for its color, one for its size
- one for its shape, one for its texture
- one for the background colour
- one for the background texture

And how VQ-VAE do this discretisation? VQ-VAE extends the VAE by adding a discrete **codebook** component to the network. The codebook is basically a list of vectors associated with a corresponding index. The output of the encoder network is compared to all the vectors in the codebook. The codebook vector closest in euclidean distance is fed to the decoder.



Here encoder-decoder network are usually simple CNN architecture which help in transforming the inputs to using activation maps.

Just like the encoder and decoder networks, the codebook vectors are learned via gradient descent where the objective is to learn codebook vectors that align to the encoder outputs and at same time learn encoder outputs that align to a codebook vector. Therefore the Loss function of a VQ-VAE has 3 terms:-

- Reconstruction Loss
- Alignment loss: Codebook vector as close to the encoder output
- Commitment loss: Encoder output to commit as much as possible to its closest codebook vector

$$\log(p(x|q(x))) + ||\text{sg}[z_e(x)] - e||_2^2 + \beta ||z_e(x) - \text{sg}[e]||_2^2$$

codebook alignment loss

codebook commitment loss

Note that If we just train VQ-VAE model by itself with the VAE objective, it will not work. Additional vector quantisation tricks are required to properly align encoder vectors with codebook vectors.

Discrete Variational Auto-encoder

If we focus on the VQ-VAE codebook vector alignment to the encoder output, we could see that it assigns probability 1 to the codebook vector nearest to the encoder's output and 0 to rest others.

$$q(\mathbf{z} = \mathbf{e}_k | \mathbf{x}) = \begin{cases} 1 & \text{if } k = \arg \min_i \|\mathbf{z}_e(\mathbf{x}) - \mathbf{e}_i\|_2 \\ 0 & \text{otherwise.} \end{cases}$$

Basically this equation is saying that the VAE's posterior distribution is deterministic. But the posterior distribution does not, in general, need to be deterministic though. In fact, you could imagine situations where the encoder might be somewhat uncertain about which codebook vector to output. And that is exactly the modification that dVAE proposes to a normal VQ-VAE.

The dVAE encoder outputs some grid of discrete latents for a given image instead of deterministically assigning 1 to only one vector in codebook space.

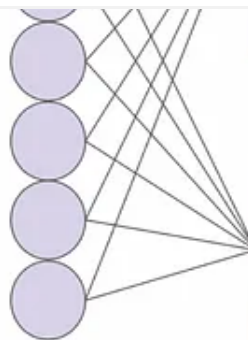
2. the encoder outputs distributions over

Latent Index	Prob
0	0.2

Open in app ↗



Search



Latent Index	Prob
0	0.4
1	0.2
2	0.15
3	0.25

	[z _{0,0} , z _{0,1} , z _{0,2} , z _{0,3} , z _{0,4} , z _{0,5} , z _{0,6} , z _{0,7} , ...]
1	[5.4, 0.65, 0.2, 4.6, 8.9, -2.43, 0.07, ...]
2	[9.78, 0.67, -3.4, 0.2, -1.0, 7.2, 13.8, ...]
3	[2.45, -8.9, 0.3, 2.04, -0.89, 19.1, 0.3, ...]

dVAE takes in an image and outputs categorical distributions over the set of codebook vectors for each latent. But the problem is that we can't back-propagate

through sampling from a categorical distribution but OpenAIs — dVAE relaxes the bottleneck — using Gumbell softmax Relaxation trick.

References:-

- [1]: <https://ml.berkeley.edu/blog/posts/vq-vae/>
- [2]: Ramesh, Aditya, et al. “Zero-shot text-to-image generation.” *International Conference on Machine Learning*. PMLR, 2021.
- [3]: Kim, Taehoon, et al. “L-Verse: Bidirectional Generation Between Image and Text.” *arXiv preprint arXiv:2111.11133* (2021).

Keep Learning, Keep Hustling.

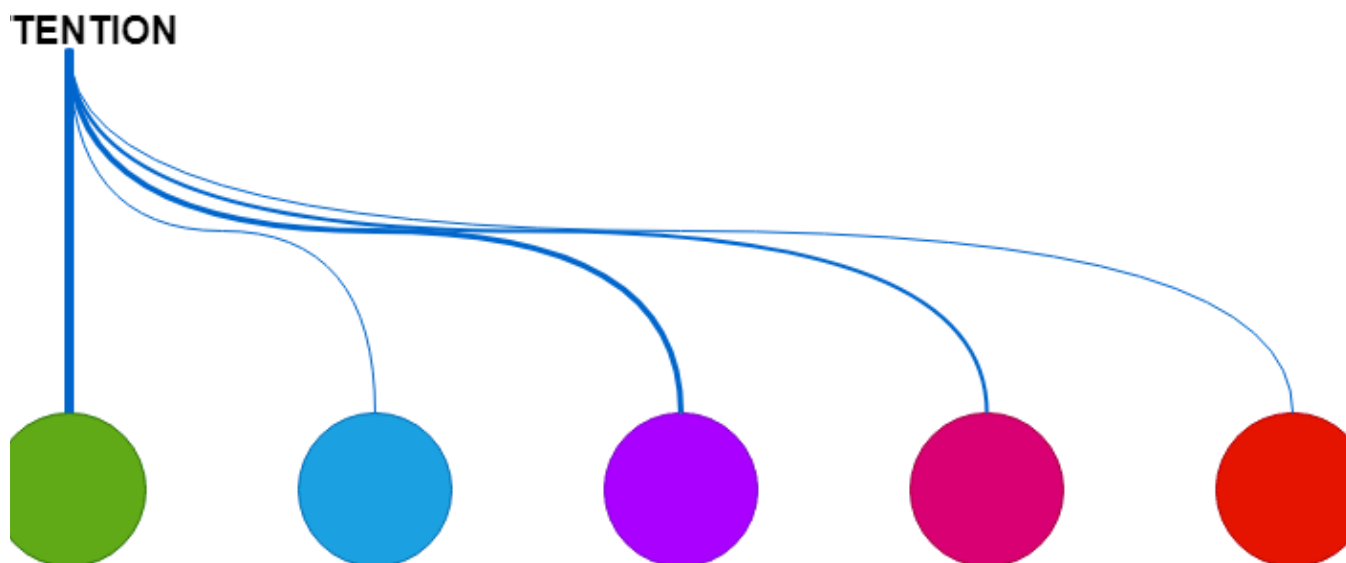
[Vae](#)[Vq Vae](#)[Dvae](#)[Autoencoder](#)[Dalle](#)[Follow](#)

Written by Nikhil Verma

921 Followers

Knowledge shared is knowledge squared | My Portfolio <https://lihkinverma.github.io/portfolio/> | My blogs are living document, updated as I receive comments

More from Nikhil Verma



Nikhil Verma

Query, Key and Value in Attention mechanism

Transformers are like bread and butter of any new research methodology and business idea developed in the field of deep learning and...



• 5 min read • Mar 26, 2022

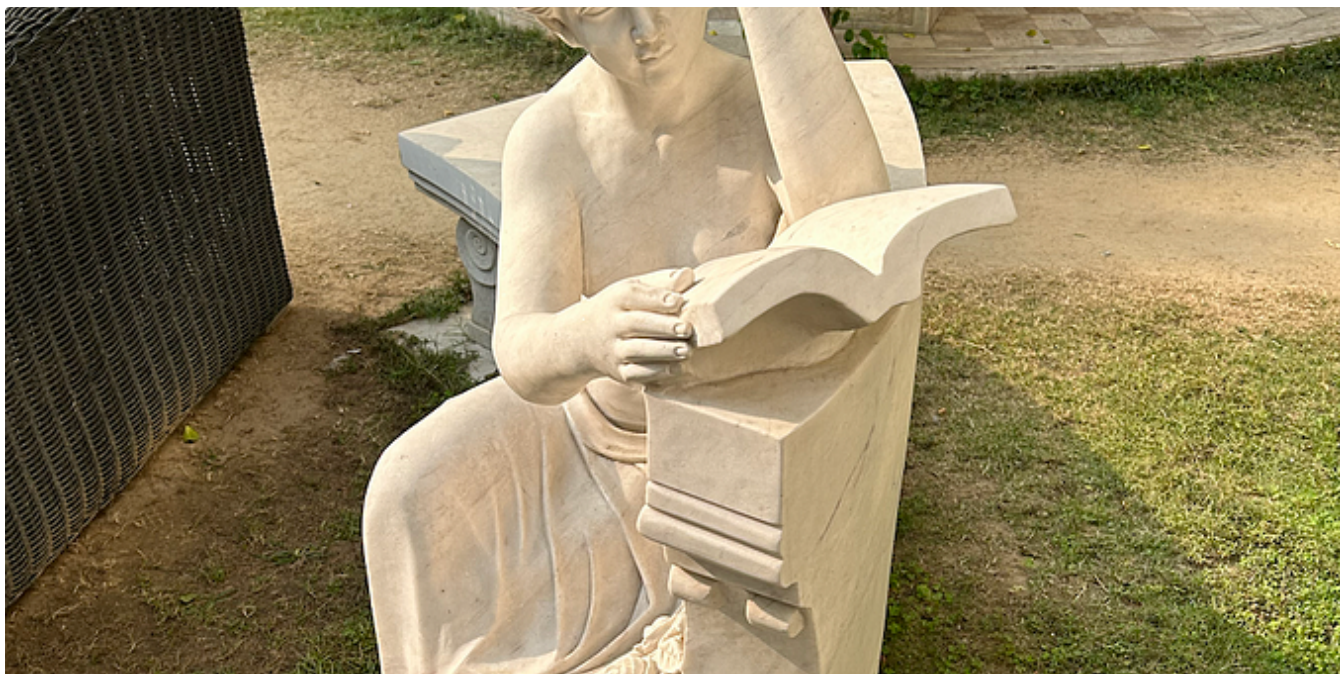


106



1





Nikhil Verma

Beyond Basics: A Comprehensive Interview Question Bank for DL, NLP, and Diffusion Models

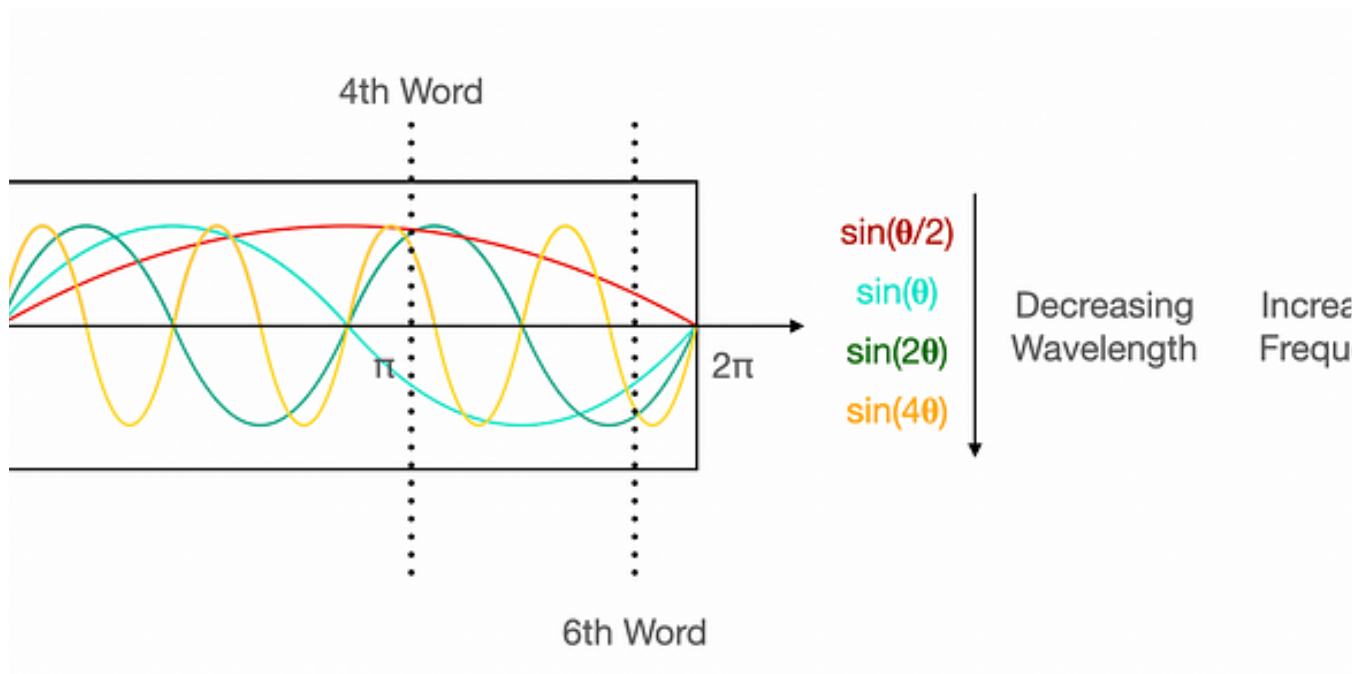
Gone are the days when a basic understanding of machine learning sufficed for tech interviews. Today, employers seek candidates who can...

🌟 · 11 min read · Dec 27, 2023



55





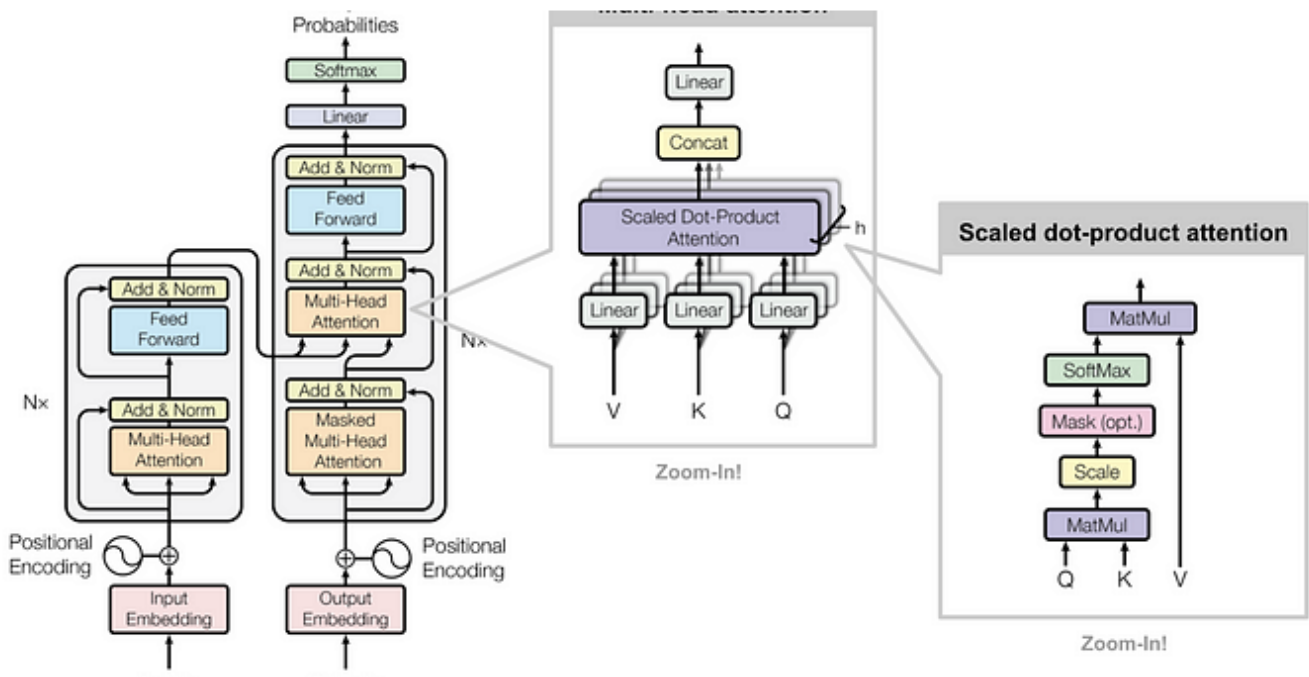
Nikhil Verma

Positional Encoding in Transformers

Transformer architecture is famous for a while having precisely designed components in itself such as Encoder-Decoder stack...

🌟 · 5 min read · Dec 28, 2022

20



Nikhil Verma

Components of Transformer Architecture

Sequence modelling is popularly done using Recurrent Neural network(RNN) or its advancements as gated RNNs or Long-short term memory(LSTM)...

★ · 2 min read · Nov 27, 2021



2



See all from Nikhil Verma

Recommended from Medium



Tiya Vaj

Variational Autoencoders

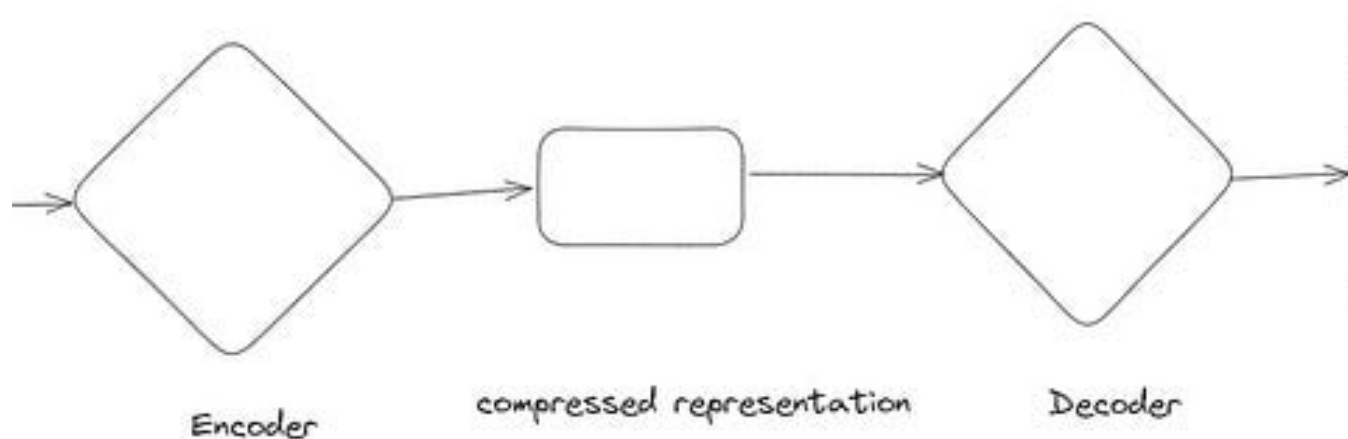
Variational Autoencoders (VAEs) are valuable in tasks that involve combining data, removing noise, and representing the inherent...

2 min read · Dec 26, 2023



1







 Wangui Waweru

AutoEncoders

Autoencoders (AE) are generative models used in unsupervised learning. AutoEncoders are neural network that compresses the input data into...

3 min read · Nov 16, 2023

 52 


Lists



Natural Language Processing

1109 stories · 581 saves



 Harshita Sharma in Accredian

Building Intuition: Variational Autoencoders (VAEs)

How Variational Autoencoders are able to generate new data

7 min read · Sep 25, 2023

 14 



 Will Badr  in Towards Data Science

Uncovering Anomalies with Variational Autoencoders (VAE): A Deep Dive into the World of...

An example use case of using Variational Autoencoders (VAE) to detect anomalies in all types of data

★ · 9 min read · Jan 17, 2023

👏 120 💬 1

🔖⁺ ⋮



 KHWAB KALRA

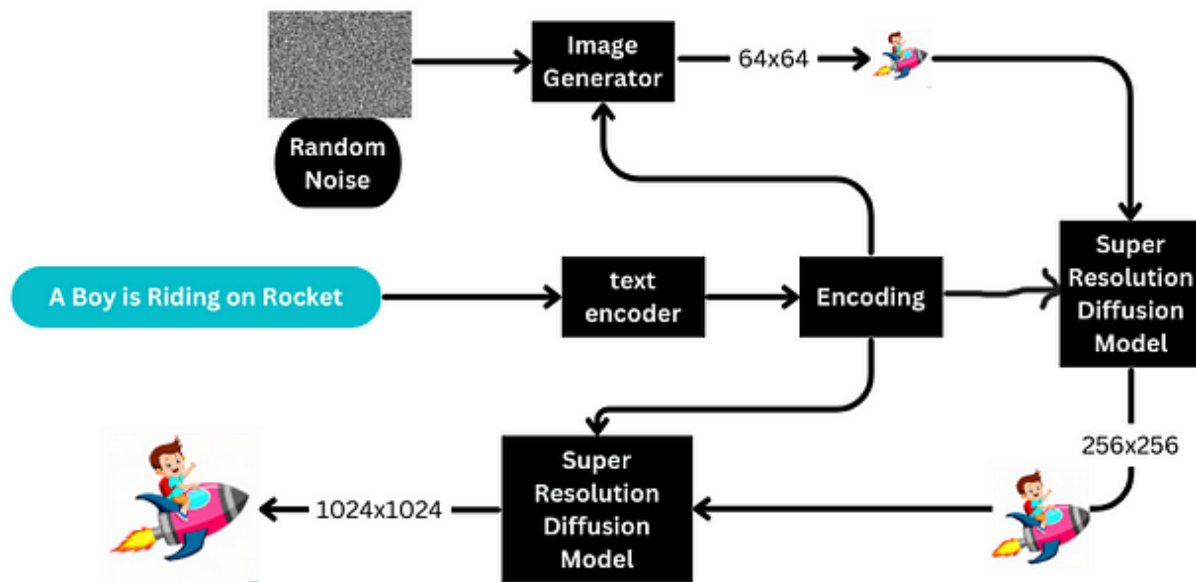
Autoencoders

Unraveling the Power of Unsupervised Representation Learning

3 min read · Jul 29, 2023

👏 14 💬

🔖⁺ ⋮





 Shyam Patel

Introduction to Diffusion Models and IMAGEN: The Magic Behind Text-to-Image Generation

Introduction to Diffusion Model

4 min read · Sep 27, 2023

 7 

See more recommendations