Harnessing Generative AI for strategic financial data synthesis

Strategy for generating synthetic data for strategic applications in financial services

Prasanth Nandanuru ,
Distinguished Engineer ,Wellsfargo

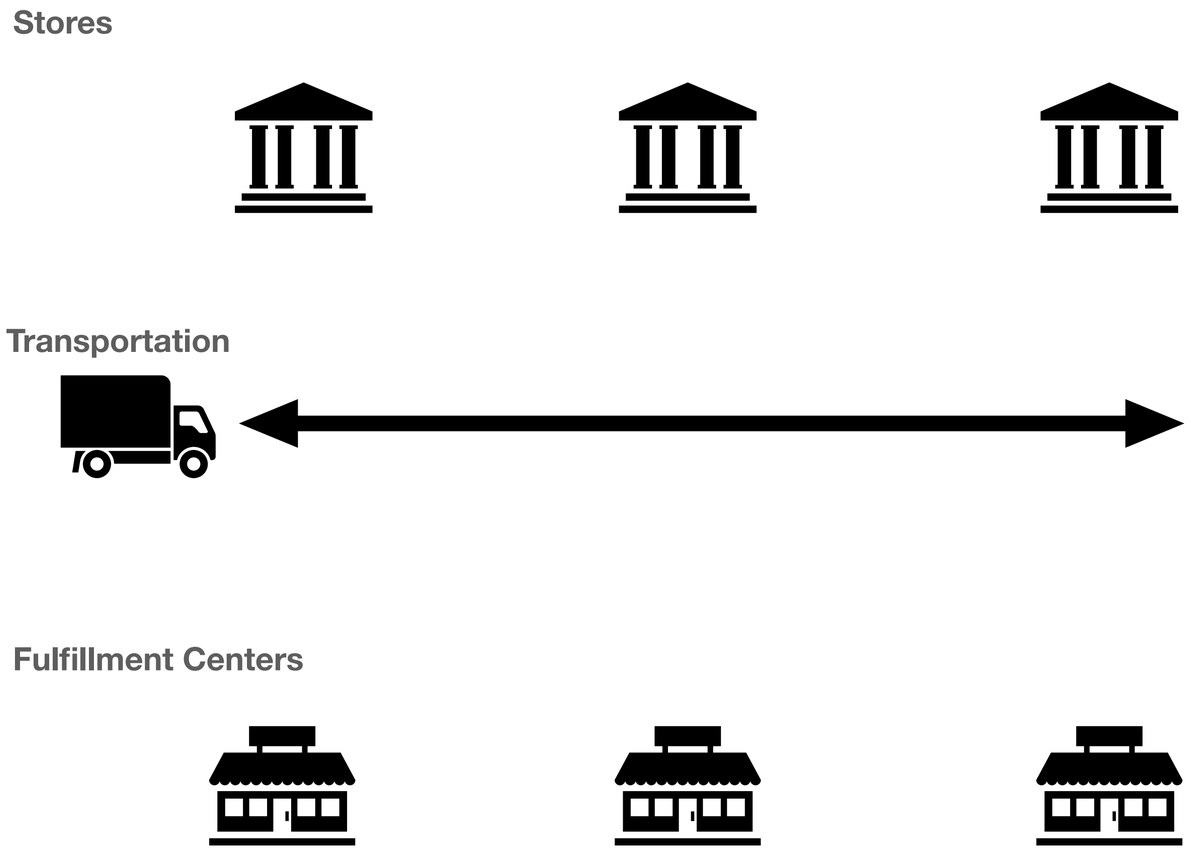
Disclaimer

Views and opinions expressed in my presentation are solely my own and do not necessarily reflect the views or opinions of my employer.While my presentation may touch on some of the work that I do at my company, it is important to note that the contents of my speech are based on my own research, thoughts, and ideas. Any statements or recommendations made during my talk are not endorsed by my employer and should not be taken as official company positions or policies.

Agenda

- Case Study : Problem Definition
- Forecasting Models:Challenges
- Synthetic data generation : Text and Images
- Privacy
- Summary

Problem Statement : Cash Recycling and Forecasting



Key Business Drivers:

- Optimal cash management -Short vs Excess
- Reduce expenses -Transportation and storage costs
- Cash Forecasting -Proactive vs Reactive
- Customer Experience

Cash Positions - Explanatory vs Observation variables

Explanatory Variables

DateTimeStamp	StoreID(AU)	Line #	GeoLocation	StoreType	Start_Cash(USD)	End_Cash(USD)
2023-09-09 09:06:53	1234	11	2pxqmv1	Prime	**20,010**	**80,000**
2023-09-09 09:08:53	4567	42	9xxqmv2	Prime	**10,000**	**4000**
2023-09-09 09:08:54	1234	17	2pxqmv1	Prime	**8000**	**9000**
2023-09-09 09:08:53	8960	8	4yxqmv4	Non	**4000**	**9000**
...						

Observation Variables

Need for Synthetic Data

Privacy and Confidentiality :PII and Regulations-GDPR/CCPA/PCI

Simulations : Planning and A/B Testing

Anomalies in real data

Data corruption

Research and Collaboration : Academia and joint collaborations

TimeSeries Data: Forecasting Methods and Challenges

UniVariate :Time vs Target variable

MultiVariate :Time + Multiple features vs Target variable

ARMA(Stationary)/ARIMA(Non Stationary)/SARIMA (Seasonality)

ARCH/GARCH -Volatility

Prophet - takes care of missing values

ARMAX (P)/LSTM (non parametric)

DART(NBEATS)

Time Series - Traditional Analysis

Time Series Analysis

Time Series is a unique type of machine learning where time plays a critical role in model predictions.

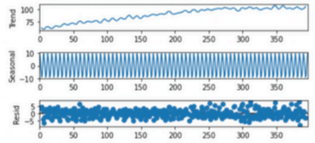
Time Series Components:

1. Trend
2. Seasonality
3. Residual

How to Describe Time Series?

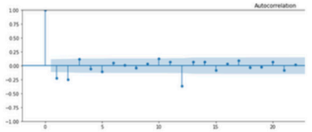
Decomposition

seasonal_decompose(df)



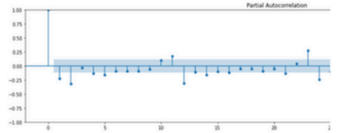
ACF

sm.graphics.tsa.plot_acf()



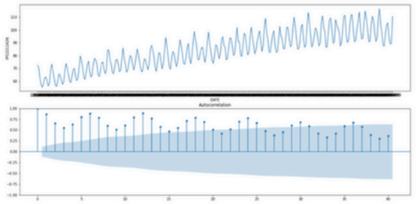
PACF

sm.graphics.tsa.plot_pacf()

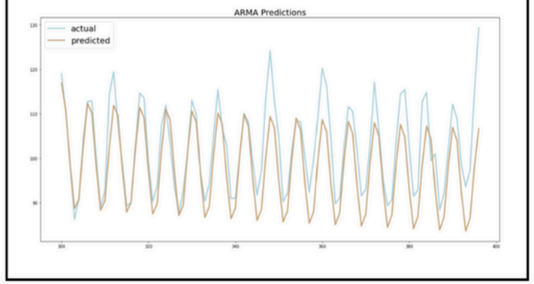


ARMA

Autoregressive + Moving Average

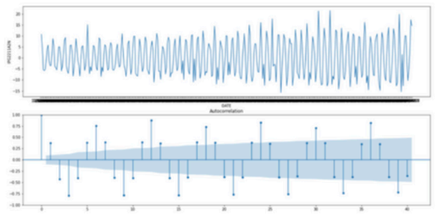


ARIMA(p, 0, q)

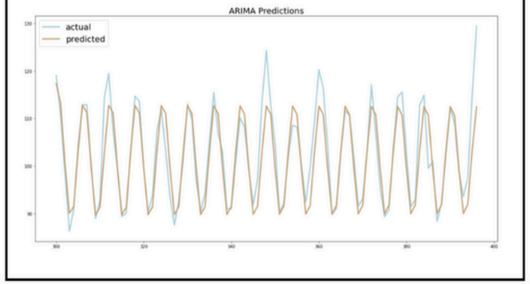


ARIMA

Autoregressive + Moving Average + Trend Differencing

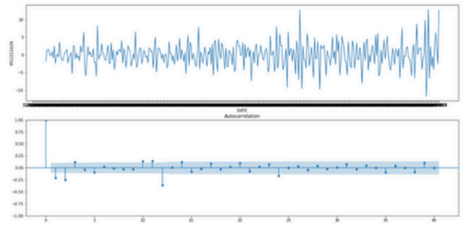


ARIMA(p, d, q)

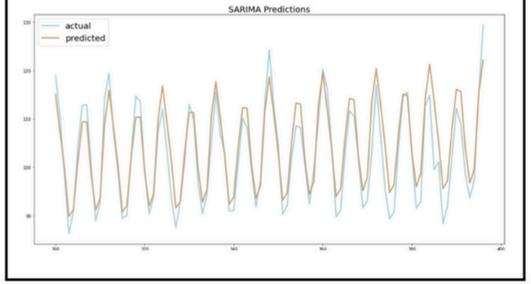


SARIMA

Autoregressive + Moving Average + Trend Differencing + Seasonality Differencing

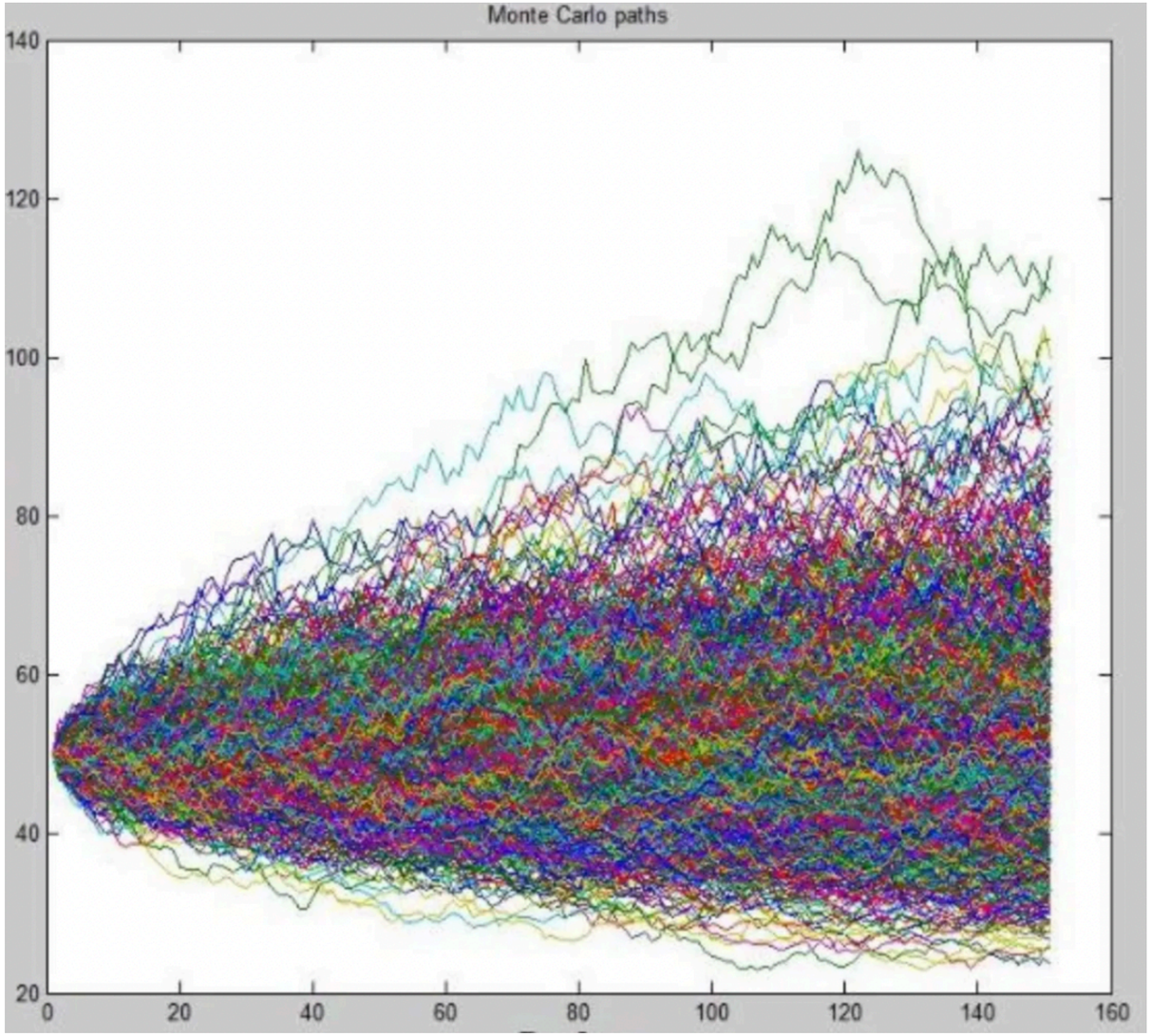


SARIMAX(p, d, q) x (P, D, Q, s)

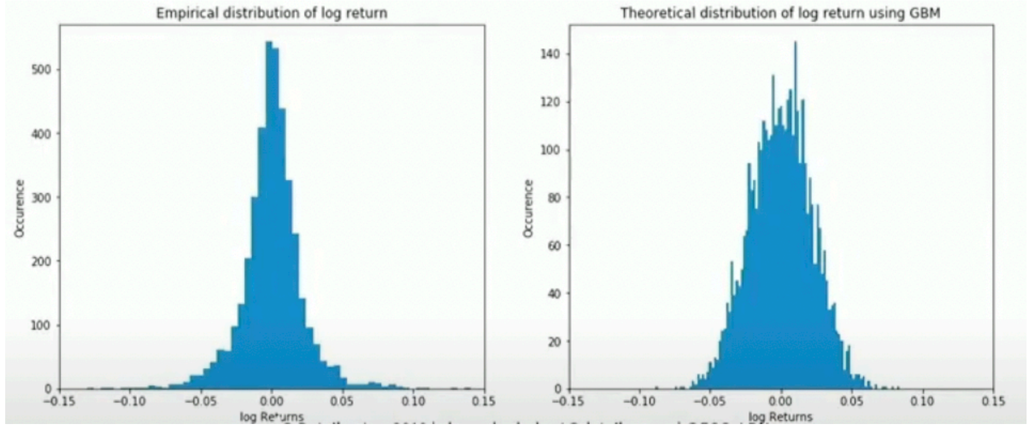


Time Series - Fat Tails Problem

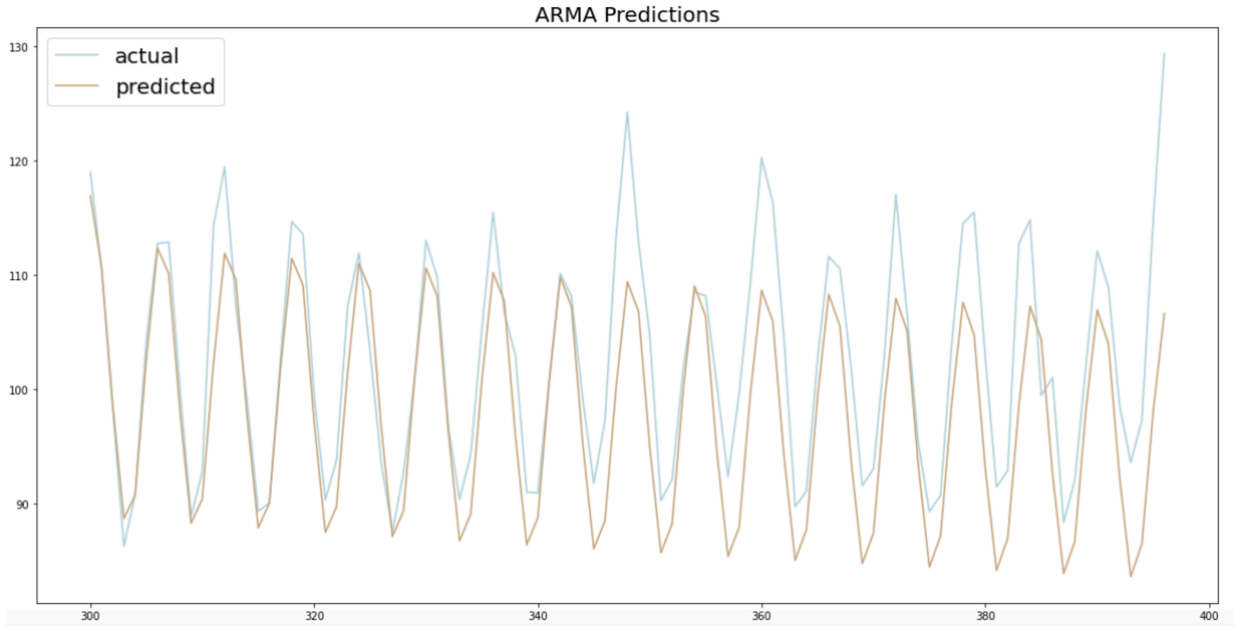
Cash Forecast



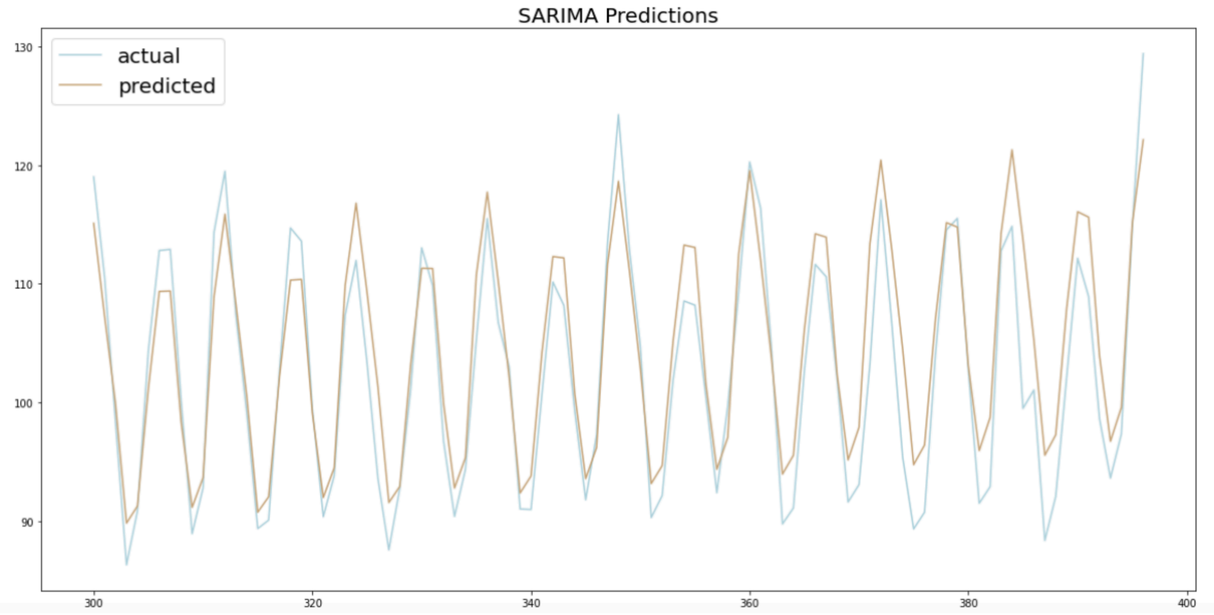
TimeSteps



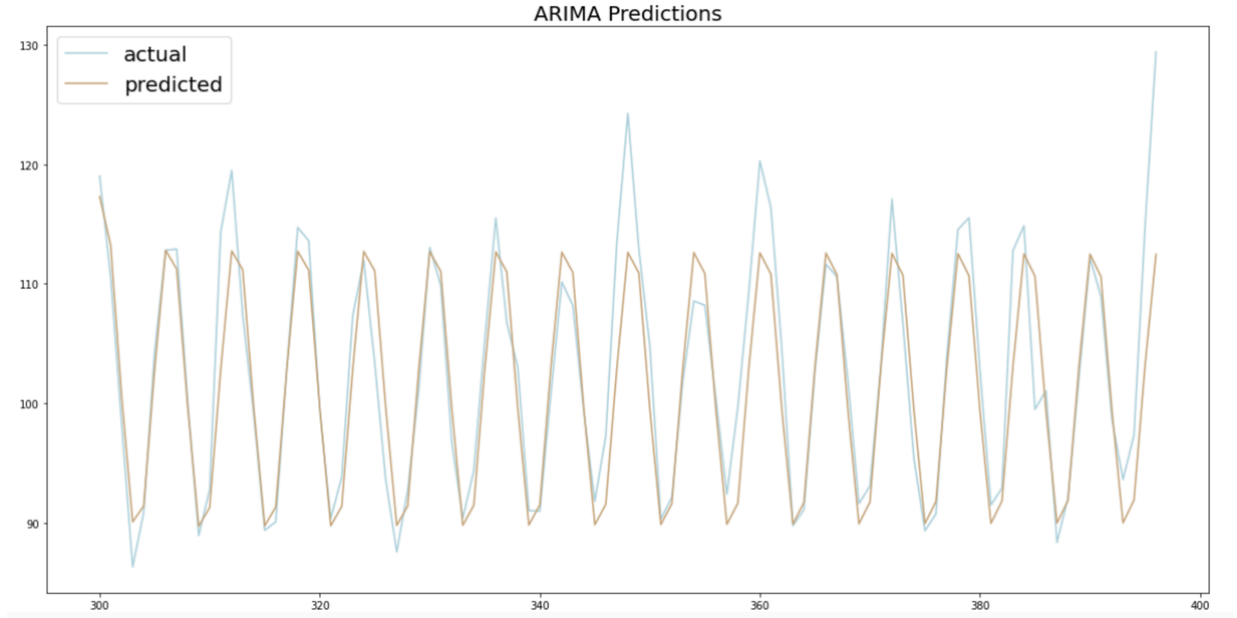
ARMA vs ARIMA vs SARIMA



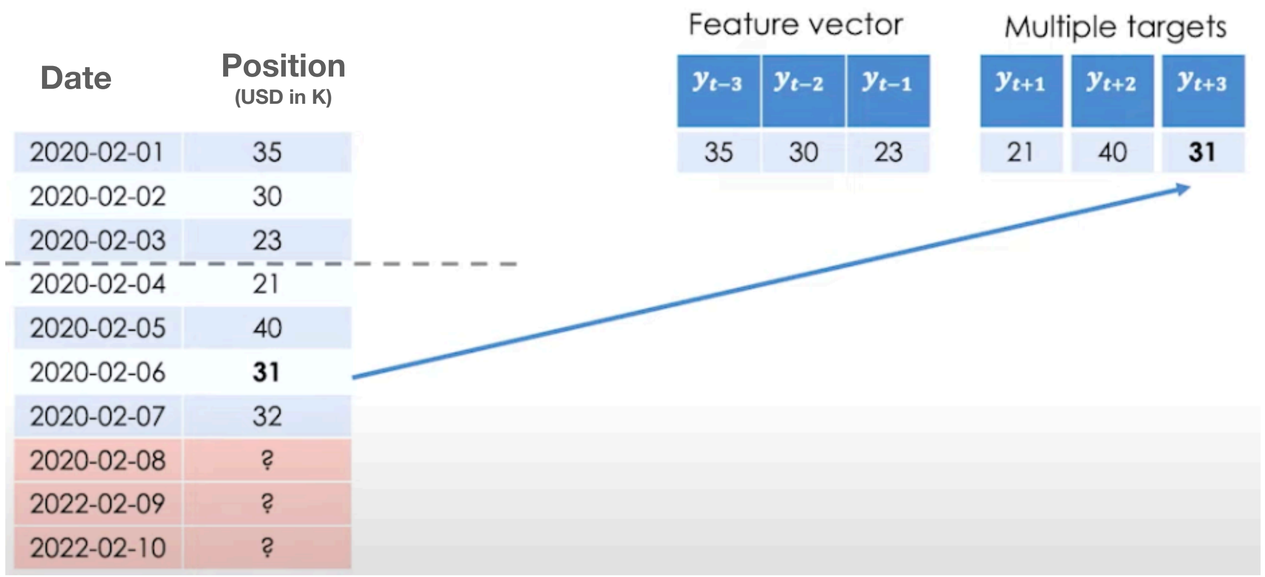
RMSE: 4.04



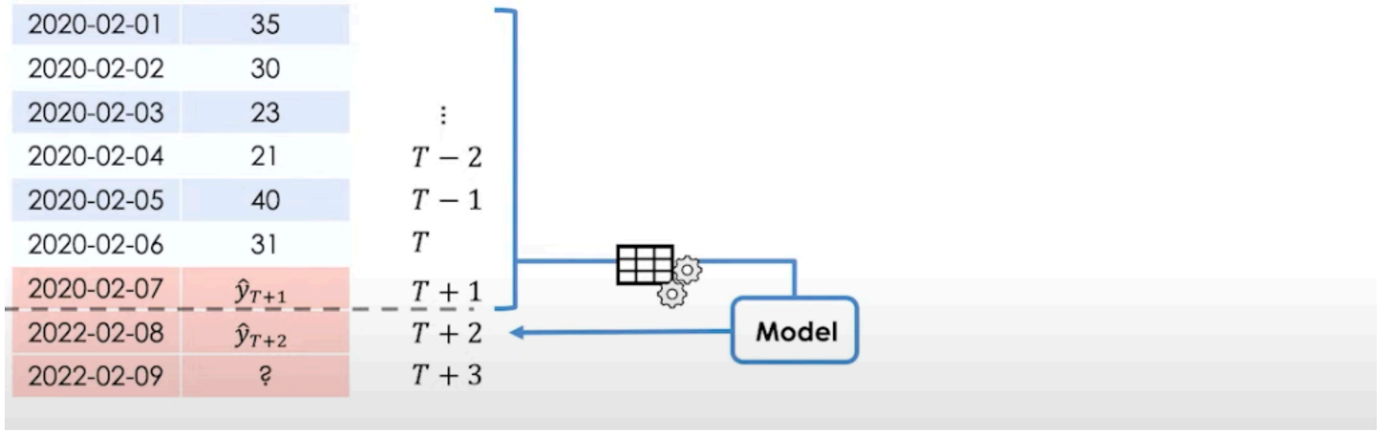
RMSE: 4.35



Time Series - Forecasting techniques

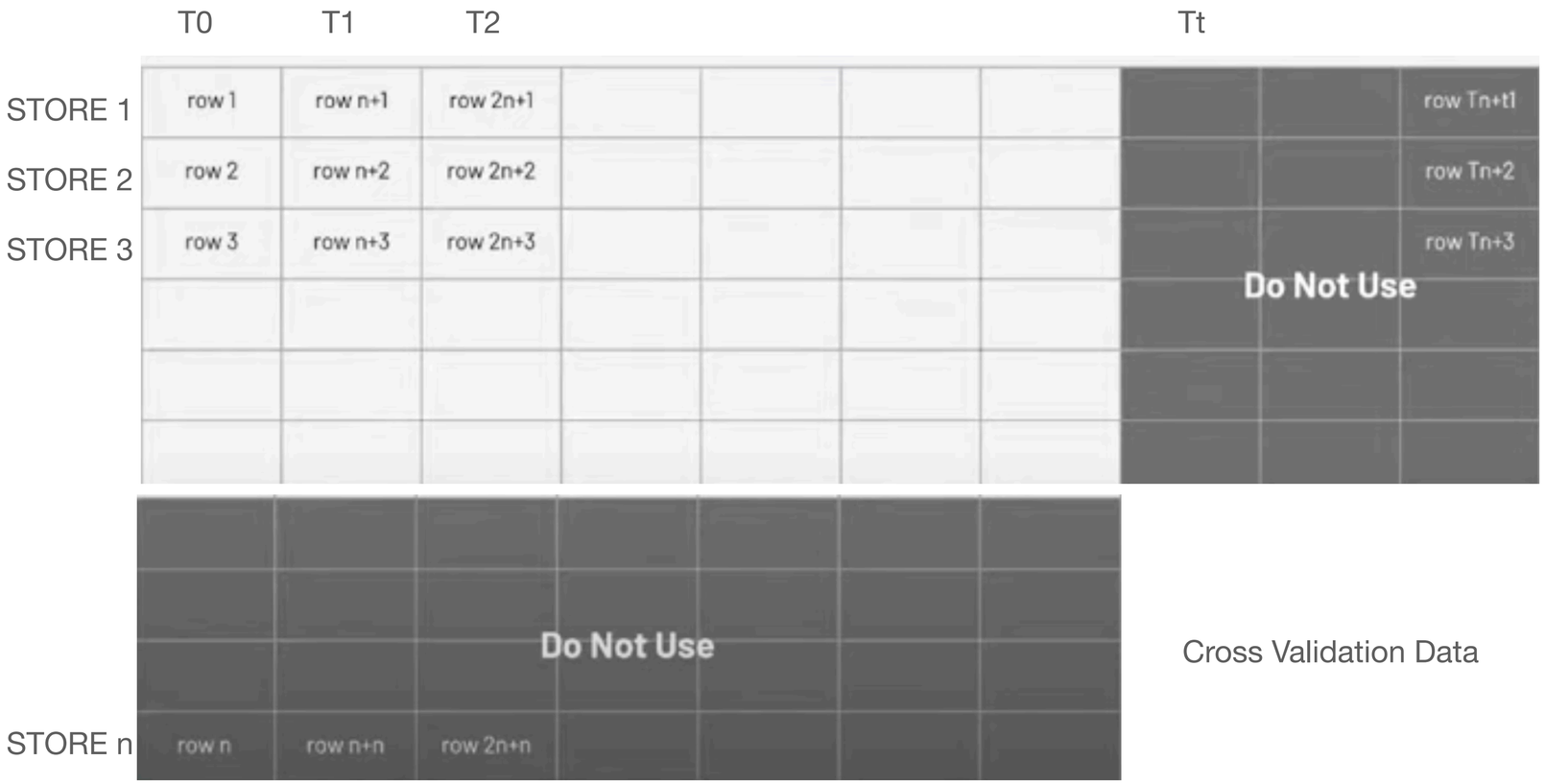


Same Features with multiple models
Each model predicts different target variable
High maintenance and inefficient

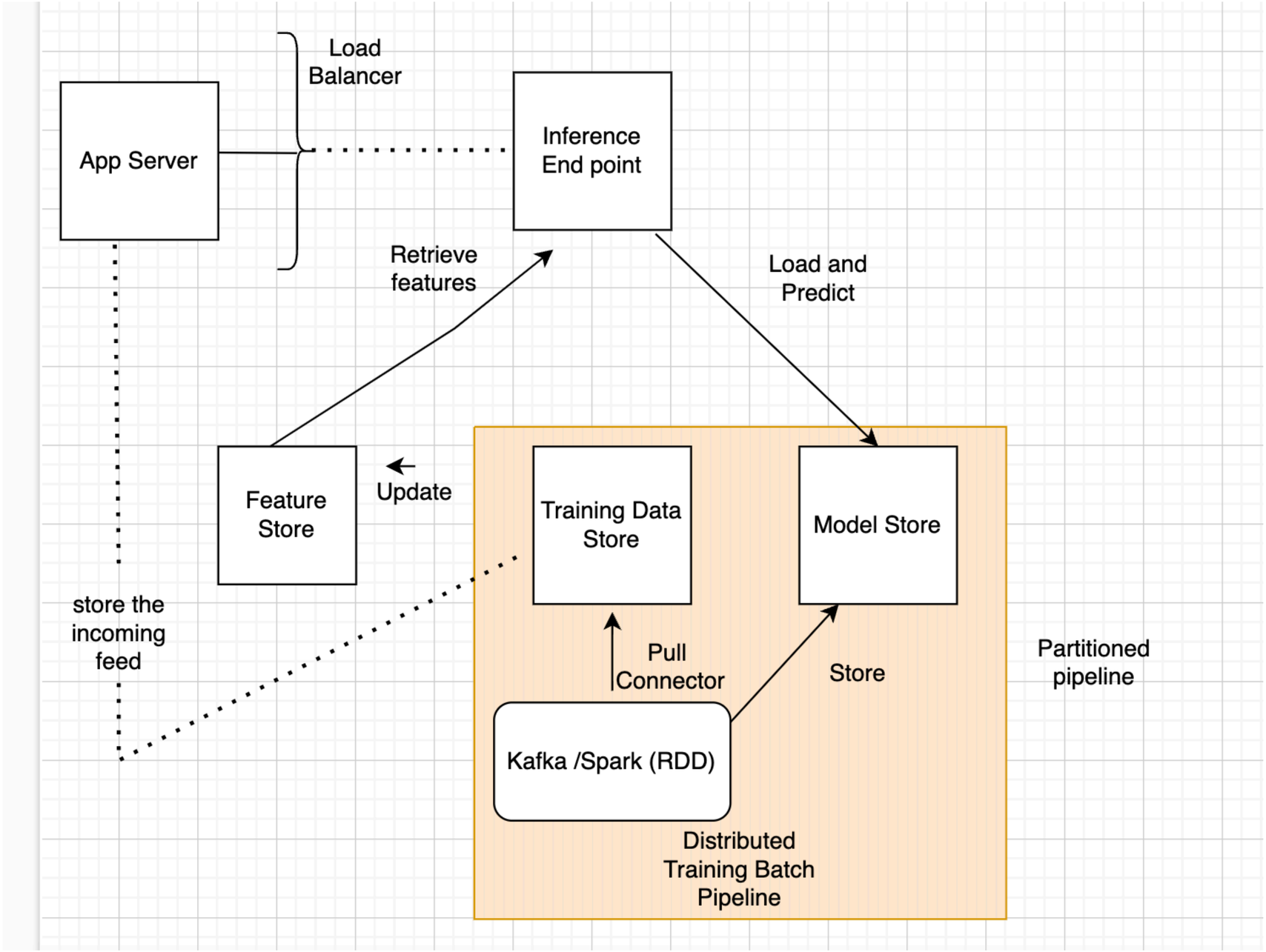


Different features
Same Model predicts target variable recursively
Efficient and low maintenance

Time Series - Cross Validation

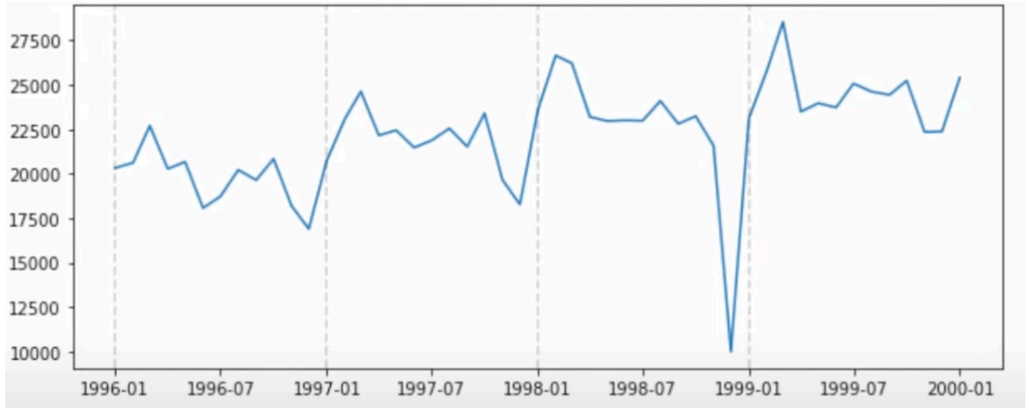


Distributed Training and Serving

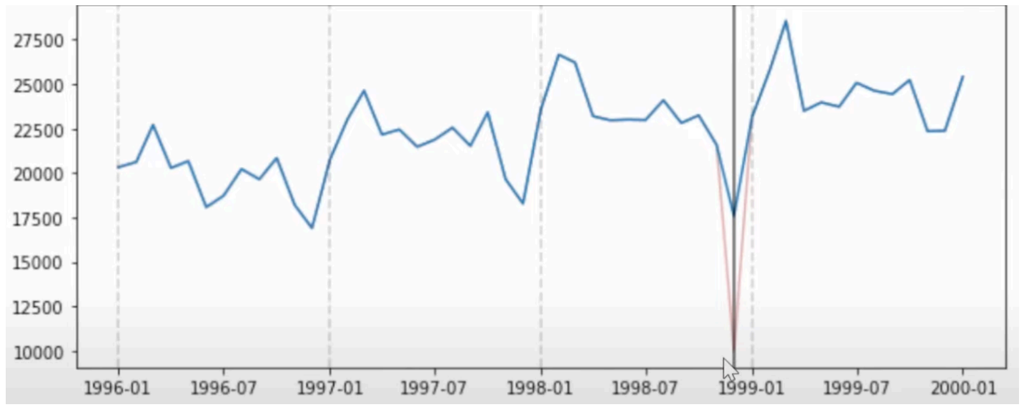


Anomaly Detection -Traditional approach

Anomaly in TimeSeries

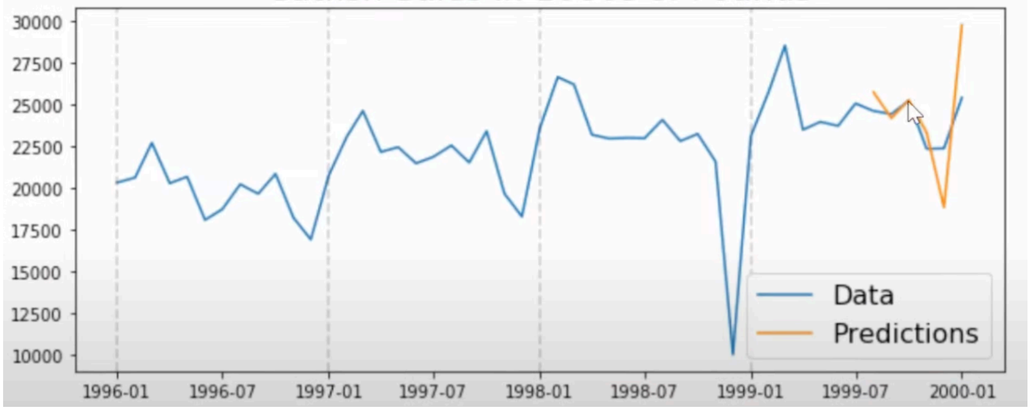


Correct the Series using : Standard Deviation using Seasonal Method



Cash Positions

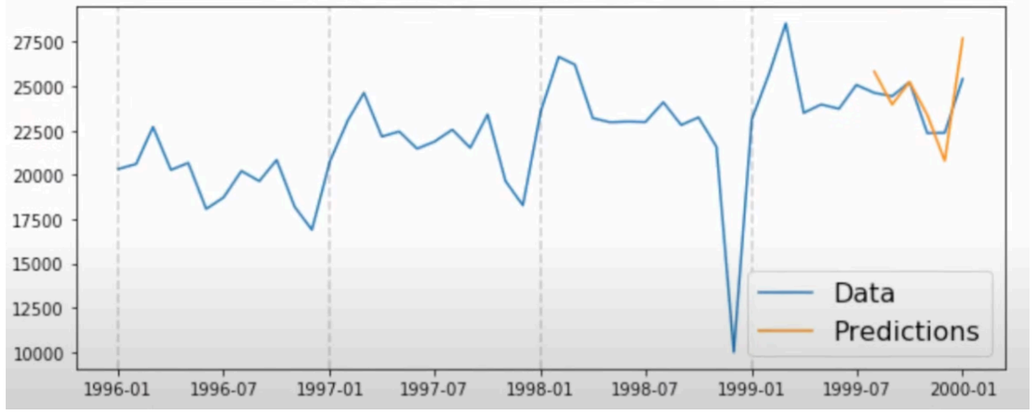
Prediction on Anomaly Timeseries



Date Window

Cash Positions

Prediction on Corrected TimeSeries



Date Window

GAN -Generative Adversarial Network

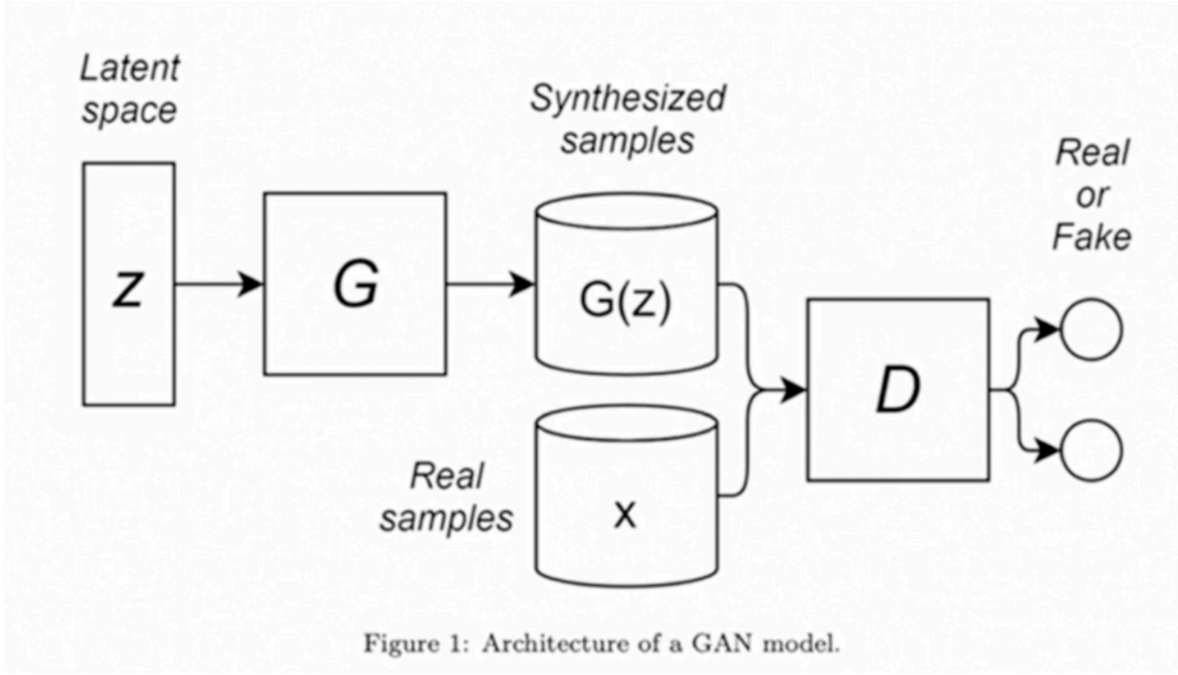
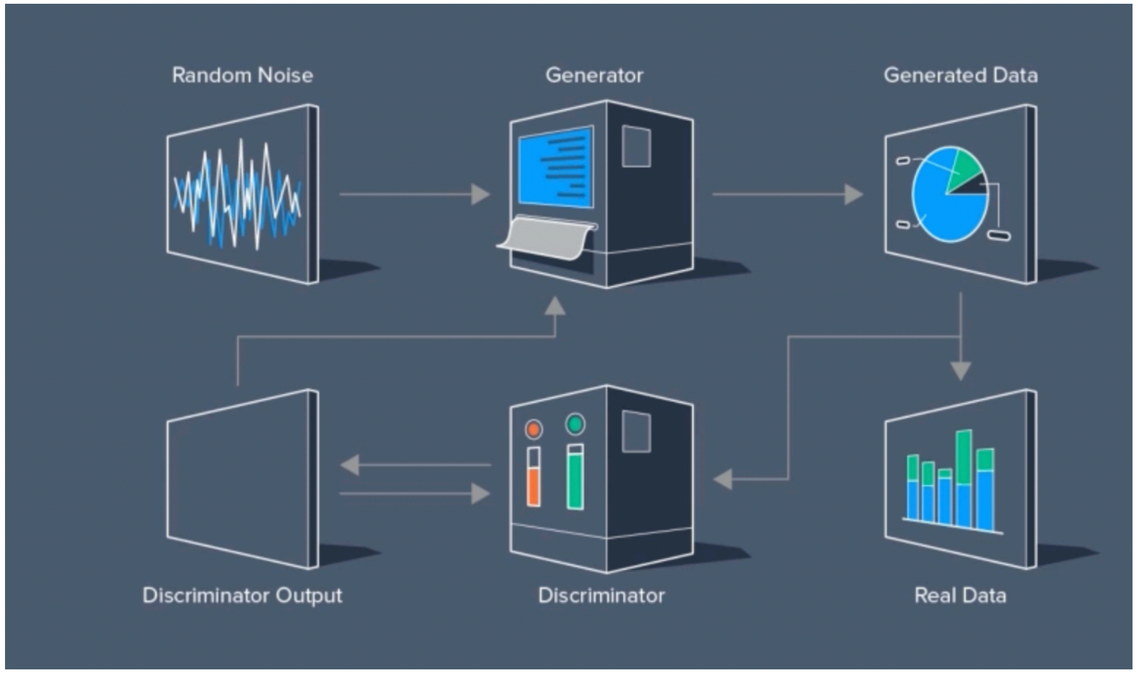


Figure 1: Architecture of a GAN model.



Two networks -Generator and Discriminator

Generator (G) model tricks the Discriminator (D) model to think generated model is real

If Discriminator(D) chooses a fake image as real , the Discriminator will adjust its model weights

If Generator(G) creates a fake image as real , the Generator will adjust its model weights

WGAN -Wasserstein GAN

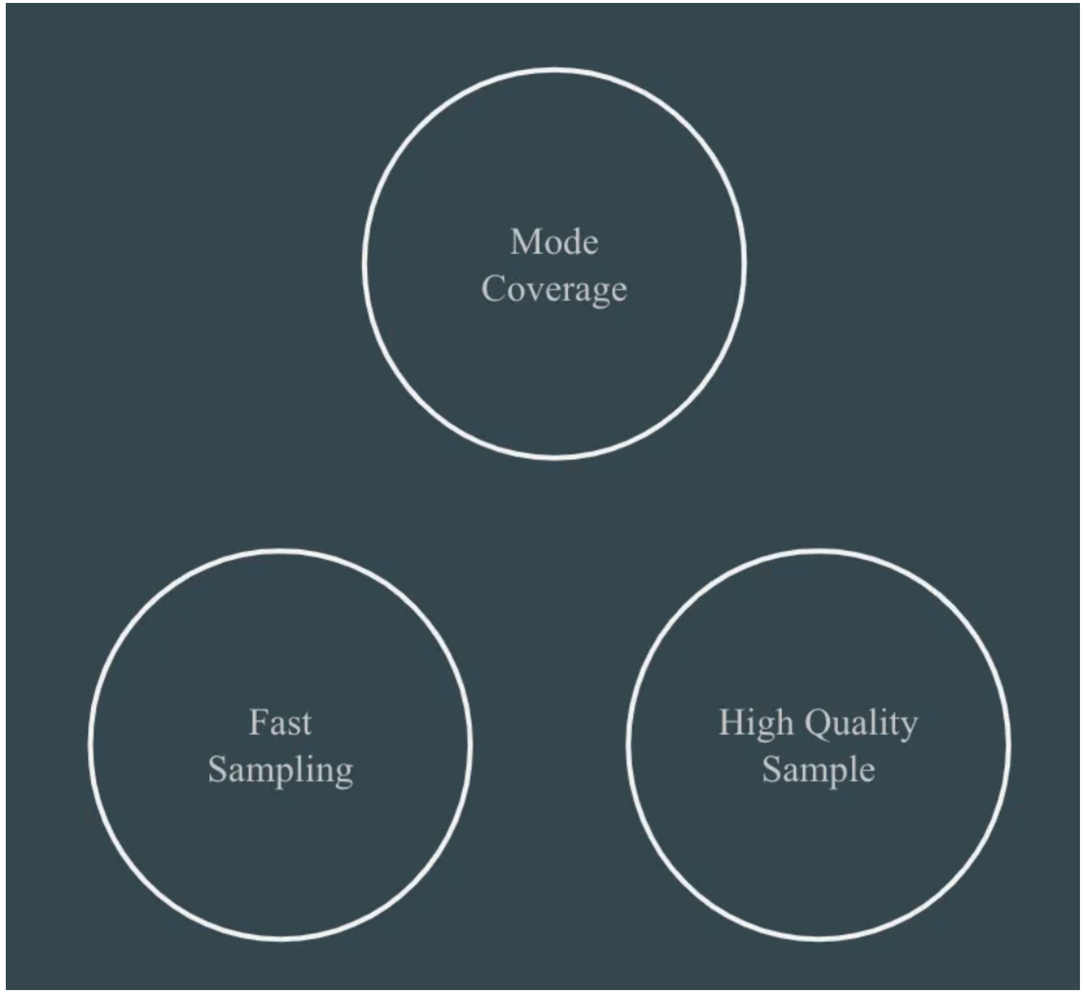
Challenges with classic GAN:

- Mode Collapse (stuck in peak of distribution)
- Loss value vs Quality of image
- Flat Gradient - Distance between distributions is very large or very small
- ARMA -Stationary

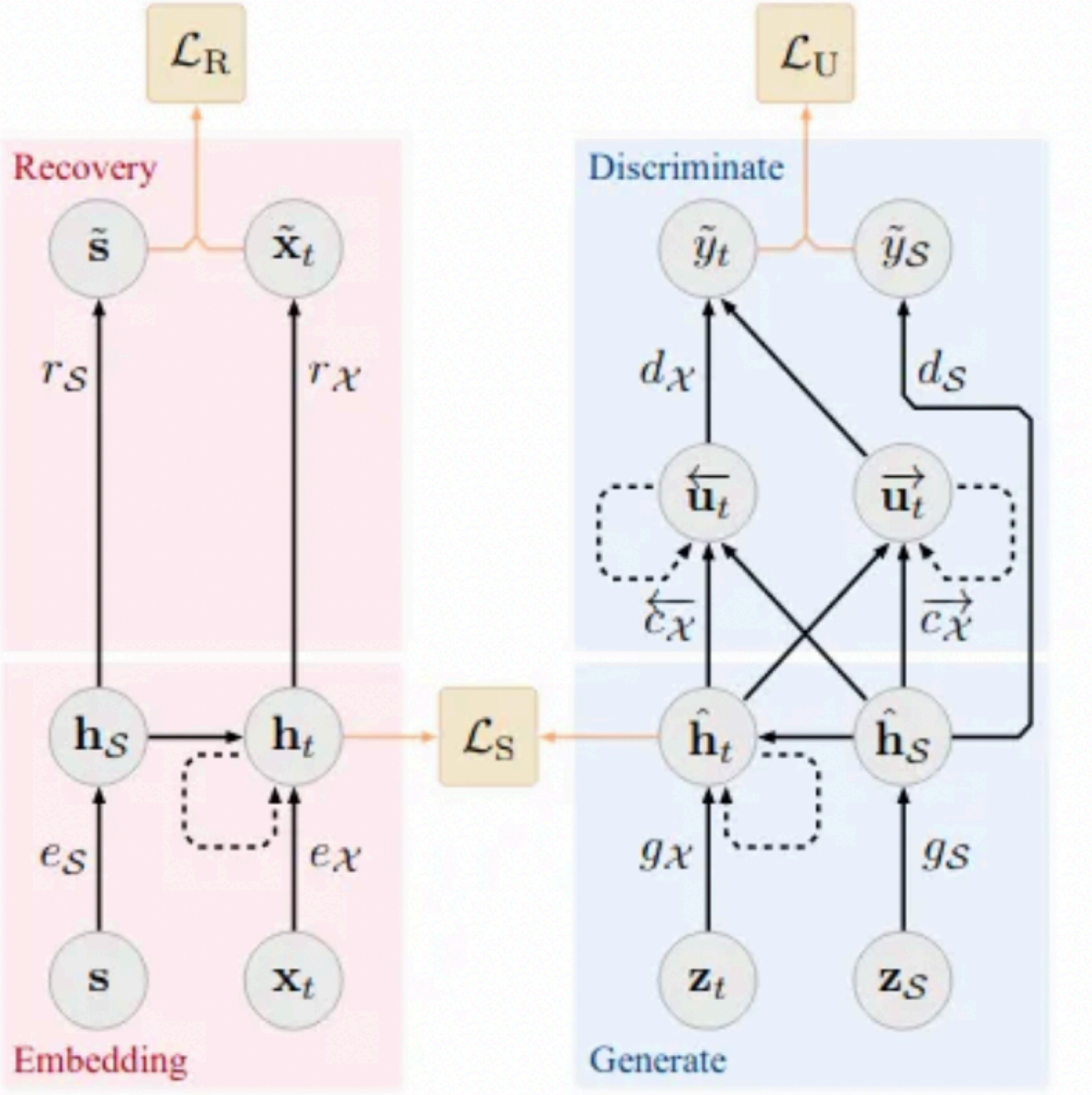
WGAN :

- Critic : No restrictions on value of critic (0 to 1)
- No more flat gradients

Generative Model Trilemma

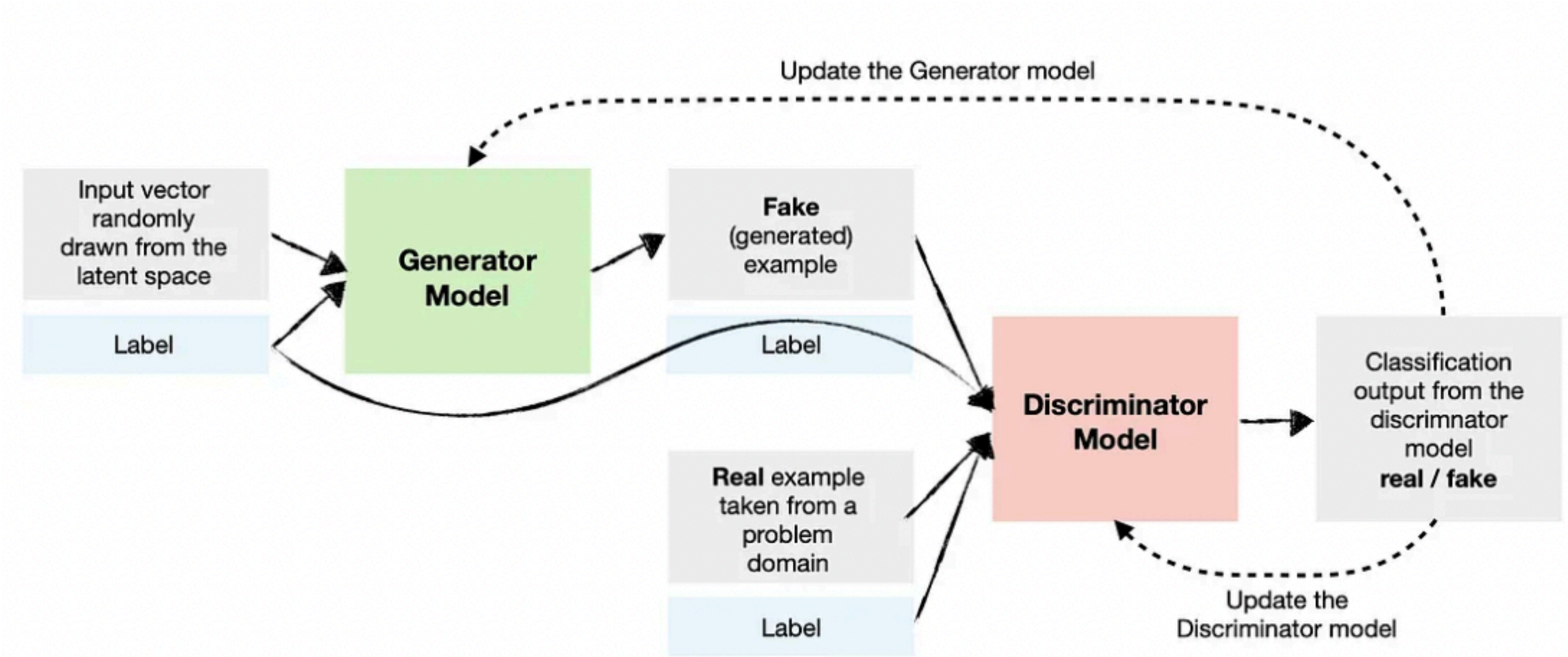


TimeGAN

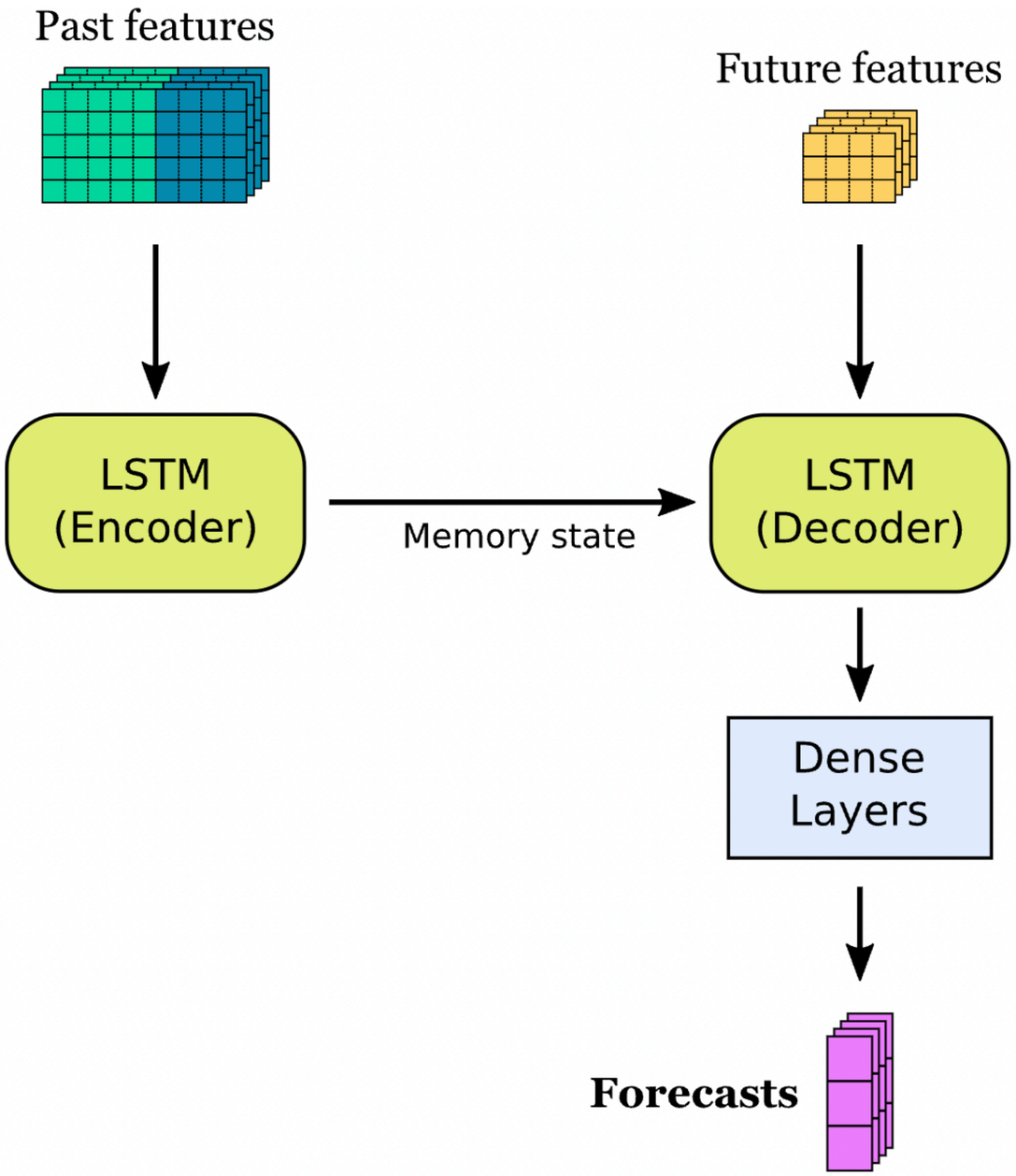


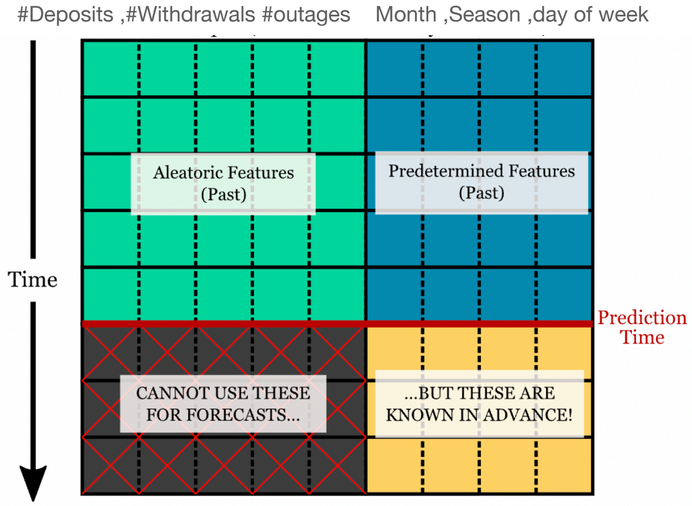
- Apart from Generator ,Discriminative models , there are Embedding and Recovery models
- 3 loss functions - Reconstruction , Supervised and Unsupervised
- Training: AutoEncoder for optimal reconstruction
- Training Supervisor to capture temporal behavior
- Training 4 models while minimizing the 3 loss functions
- Less sensitive to hyper parameters changes and more stable training process

Conditional GAN - Multiple inputs

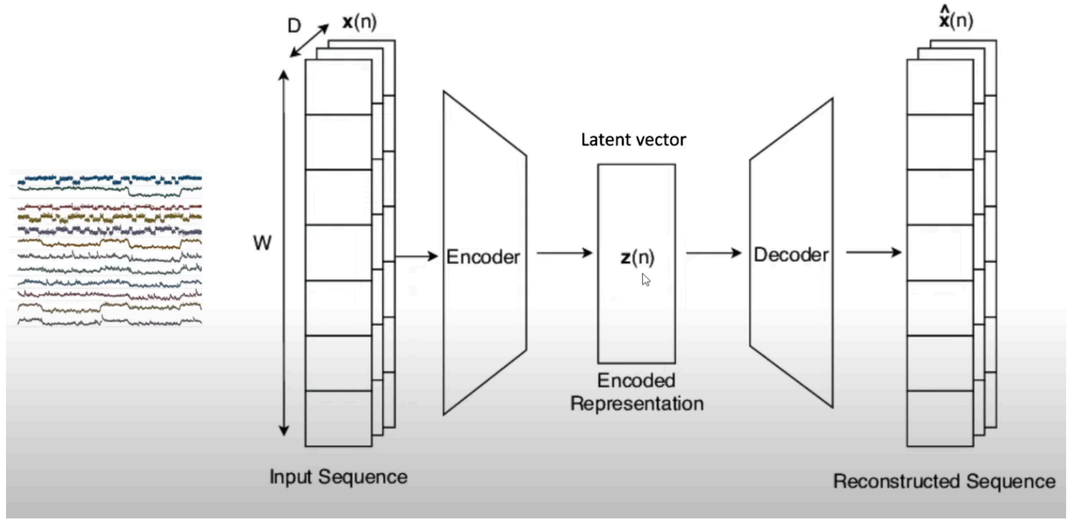


LSTM -Long Short Term Memory Models





LSTM -Anomaly Detection



Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 30, 128)	66560
lstm_1 (LSTM)	(None, 64)	49408
repeat_vector (RepeatVector)	(None, 30, 64)	0
lstm_2 (LSTM)	(None, 30, 64)	33024
lstm_3 (LSTM)	(None, 30, 128)	98816
time_distributed (TimeDistri	(None, 30, 1)	129

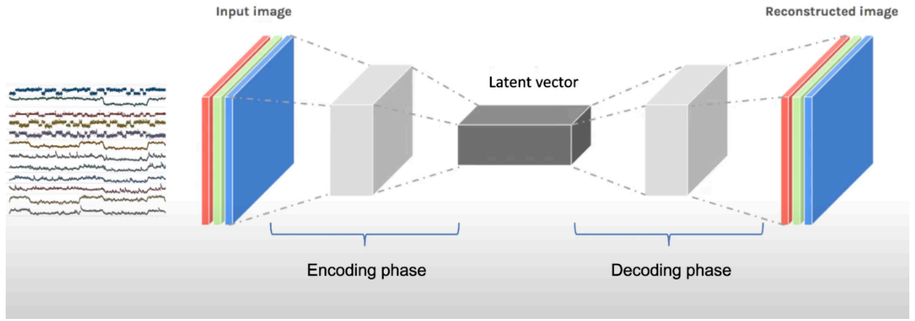
Total params: 247,937
Trainable params: 247,937
Non-trainable params: 0

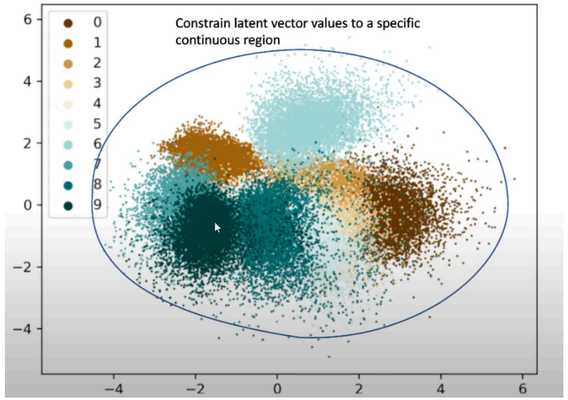
Input: (N, 30, 1)

```
model = Sequential()
model.add(LSTM(128, activation='relu', input_shape=(trainX.shape[1], trainX.shape[2]), return_sequences=True))
model.add(LSTM(64, activation='relu', return_sequences=False))
model.add(RepeatVector(trainX.shape[1]))
model.add(LSTM(64, activation='relu', return_sequences=True))
model.add(LSTM(128, activation='relu', return_sequences=True))
model.add(TimeDistributed(Dense(trainX.shape[2])))

model.compile(optimizer='adam', loss='mse')
```

Variational AutoEncoders -Reparameterisation Trick





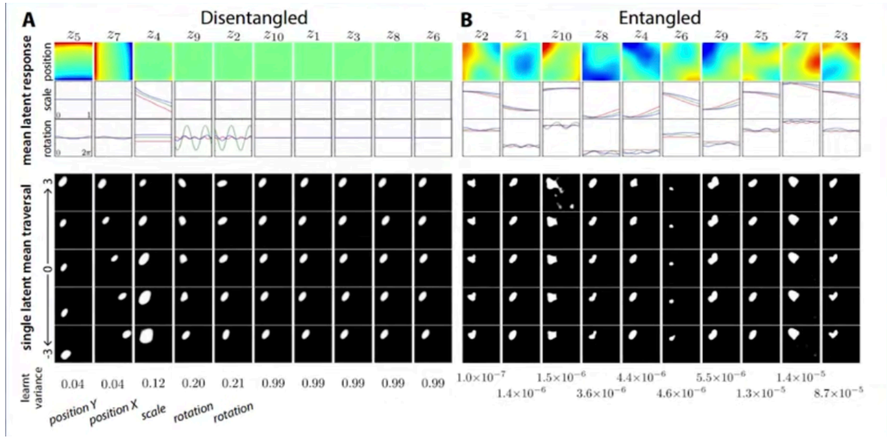
Learned parameters (during back propagation)

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$$

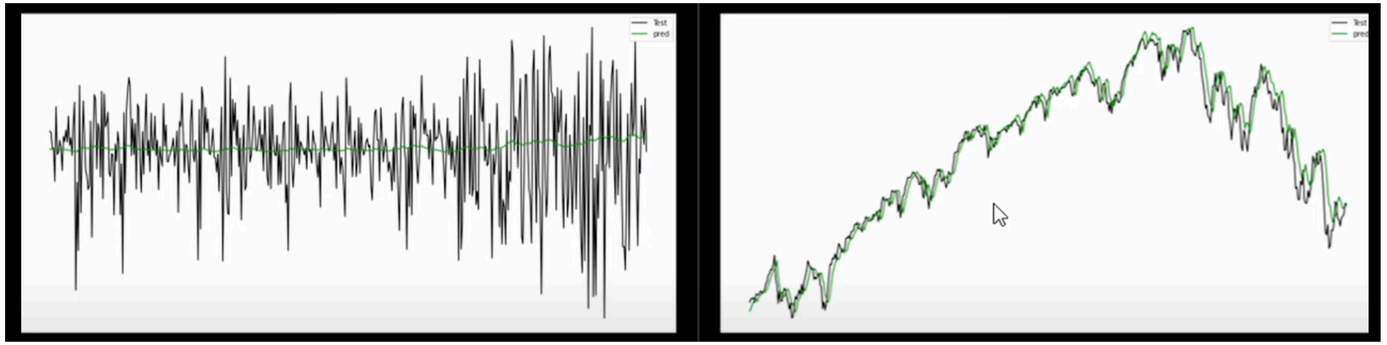
Standard normal distribution
Fixed space where we randomly sample
Not learned during back propagation

Disentangled Variational AutoEncoders



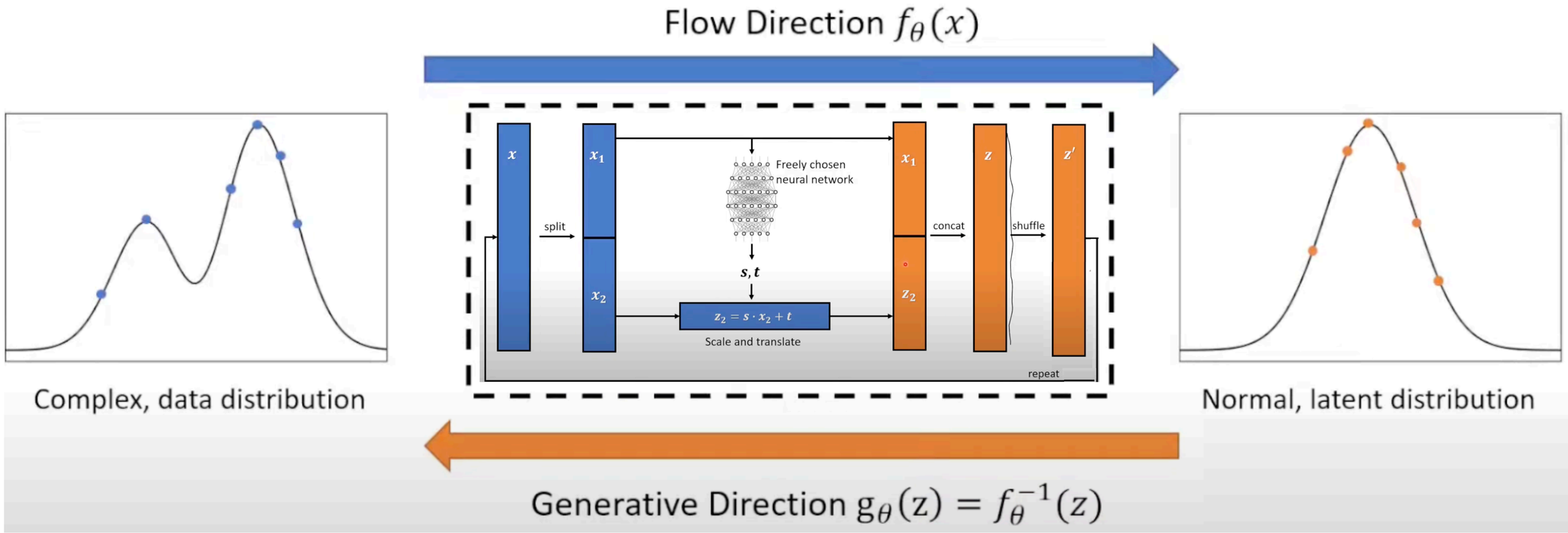
$$\mathcal{L}(\theta, \phi; \mathbf{x}, \mathbf{z}, \beta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta D_{KL}\left(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\right)$$

LSTM Tricks : Price Movements vs. Close price in Trading securities



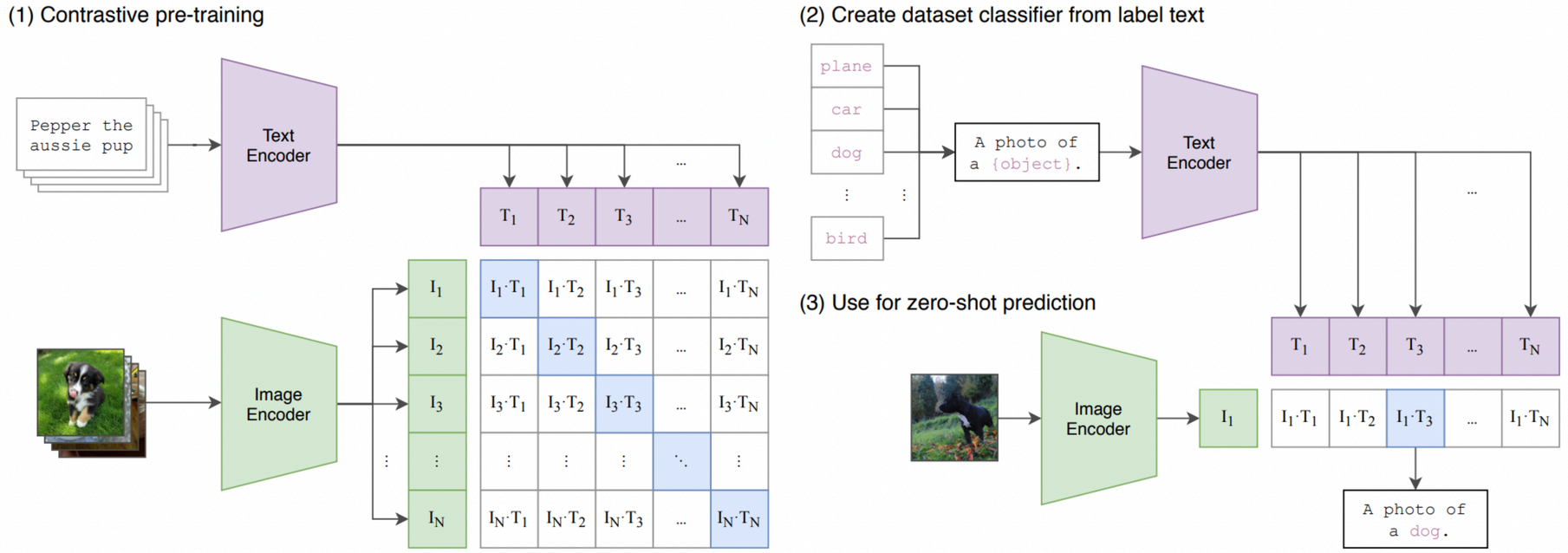
Adj Close	RSI	EMAF	EMAM	EMAS	Target	TargetClass	TargetNextClose
789.760010	46.877256	795.406526	775.179523	762.677734	-2.690002	0	787.179993
787.179993	44.575540	794.623046	775.417156	763.002267	6.260010	1	793.440002
793.440002	50.849388	794.510376	775.774044	763.405416	7.019958	1	801.599976
801.599976	57.558969	795.185576	776.285448	763.911304	4.369995	1	805.039978

Flow based models



CLIP MultiModal architecture -Generate synthetic data based on text and image

CLIP: Contrastive Language-Image Pre-Training



VQGAN - Generate Synthetic Paper Receipts

Fig.a

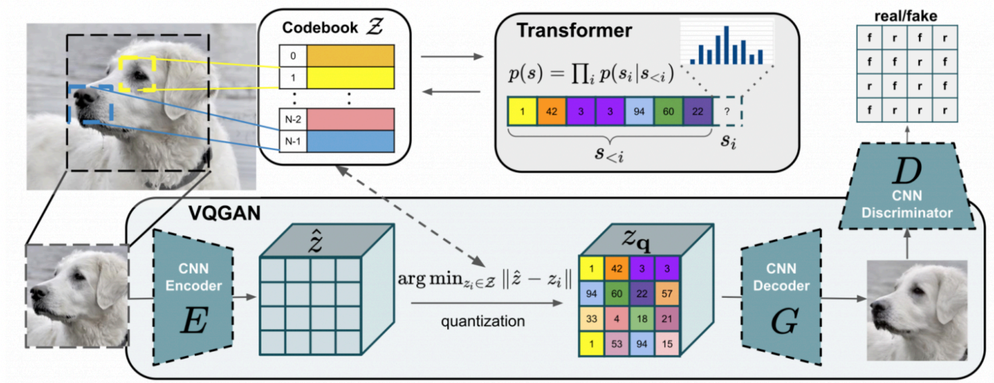
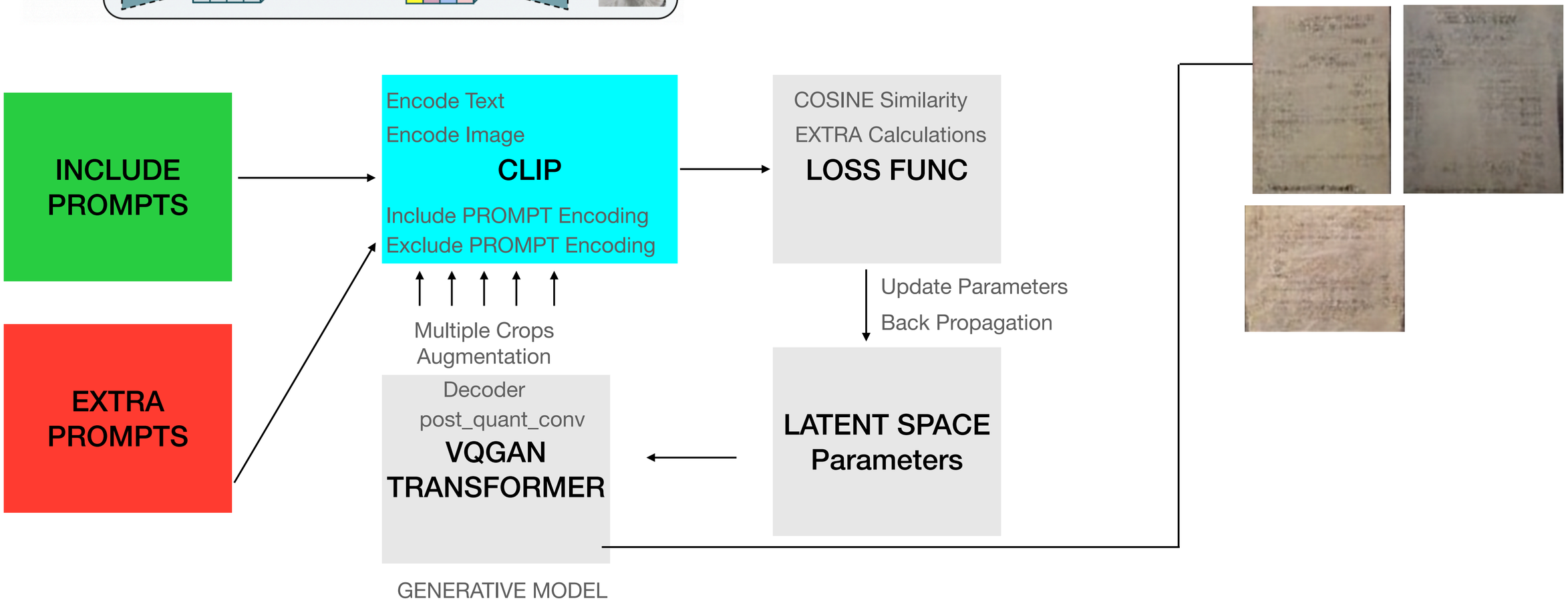
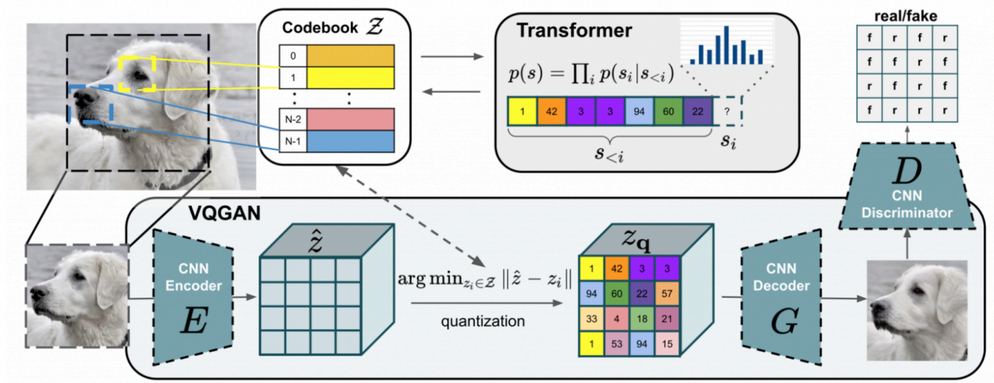


Fig.b



VQGAN - Generate QR codes

Fig.a



Text Prompt: Ancient Village

Image

EXTRA PROMPTS

Encode Text
Encode Image
CLIP
Include PROMPT Encoding
Exclude PROMPT Encoding

Multiple Crops
Augmentation

Decoder
post_quant_conv
VQGAN TRANSFORMER

GENERATIVE MODEL

COSINE Similarity
EXTRA Calculations
LOSS FUNC

Update Parameters
Back Propagation

LATENT SPACE Parameters

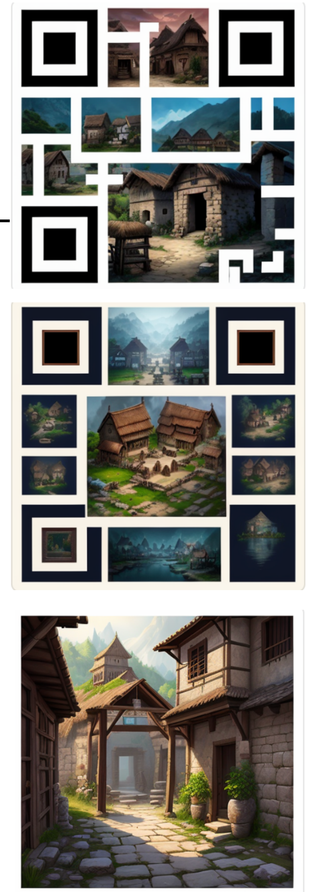


Fig.b

VQGAN - Generate scannable QR codes based on famous locations

Prompt:



Grand Canyon

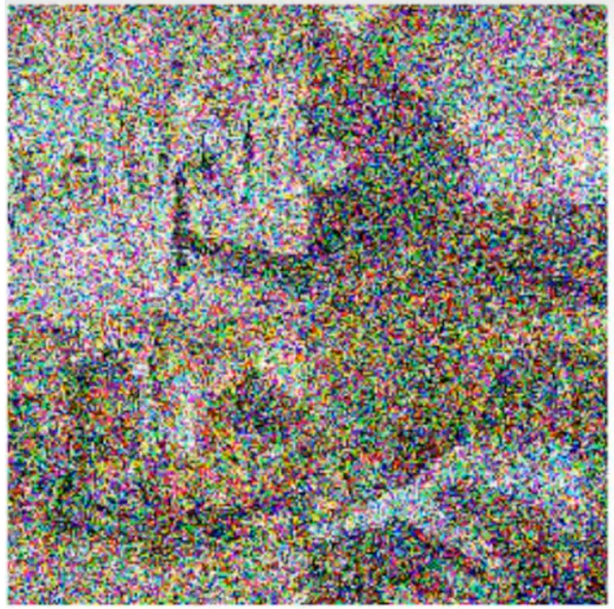


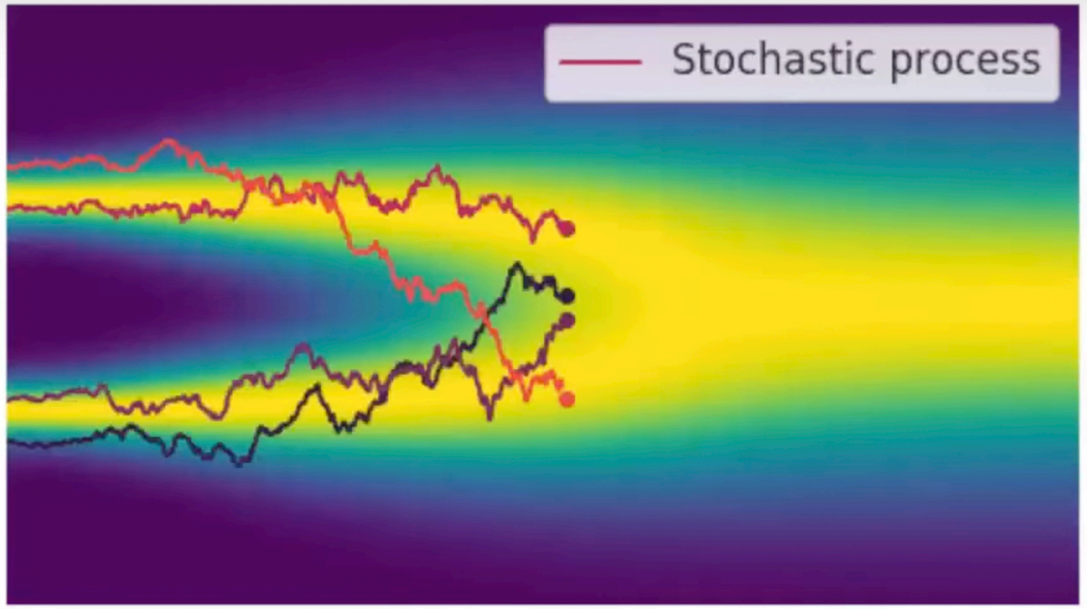
Mount Rushmore



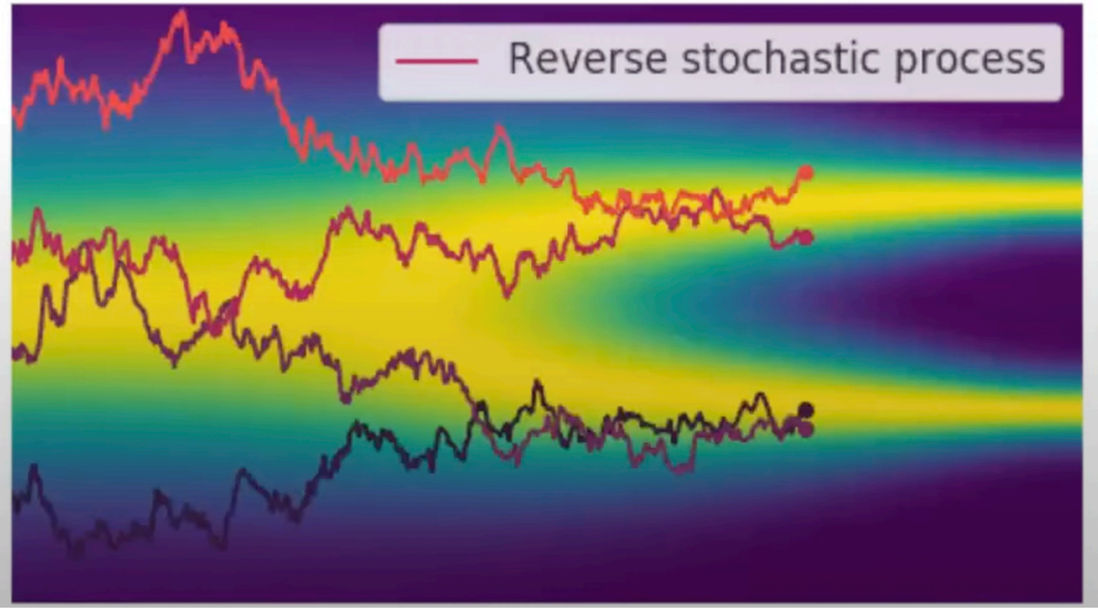
Golden Gate bridge

Diffusion Models :Forward and Reverse Process



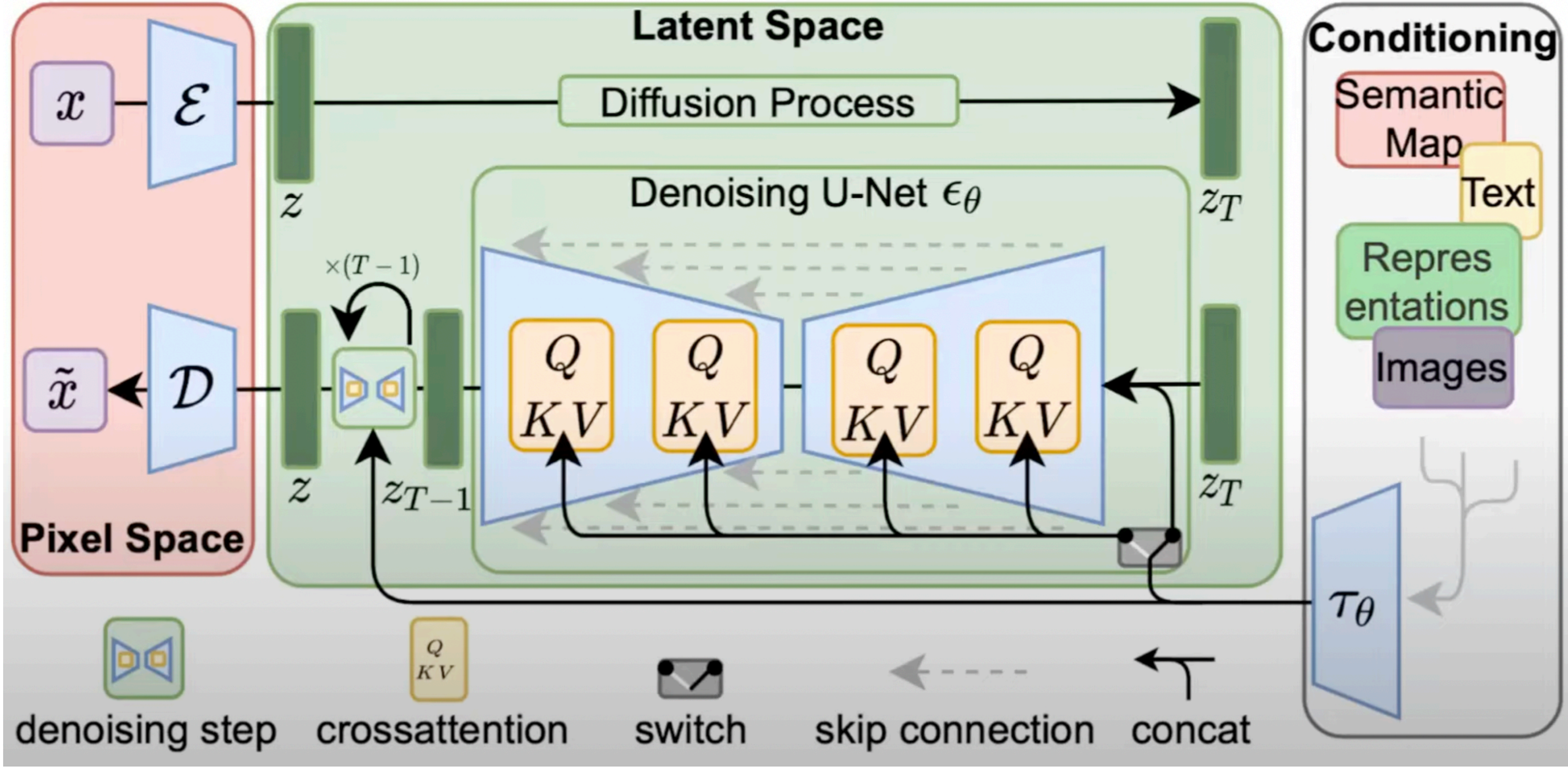


Forward process: converting the image distribution to pure noise



Reverse process: sampling from the image distribution, starting with pure noise

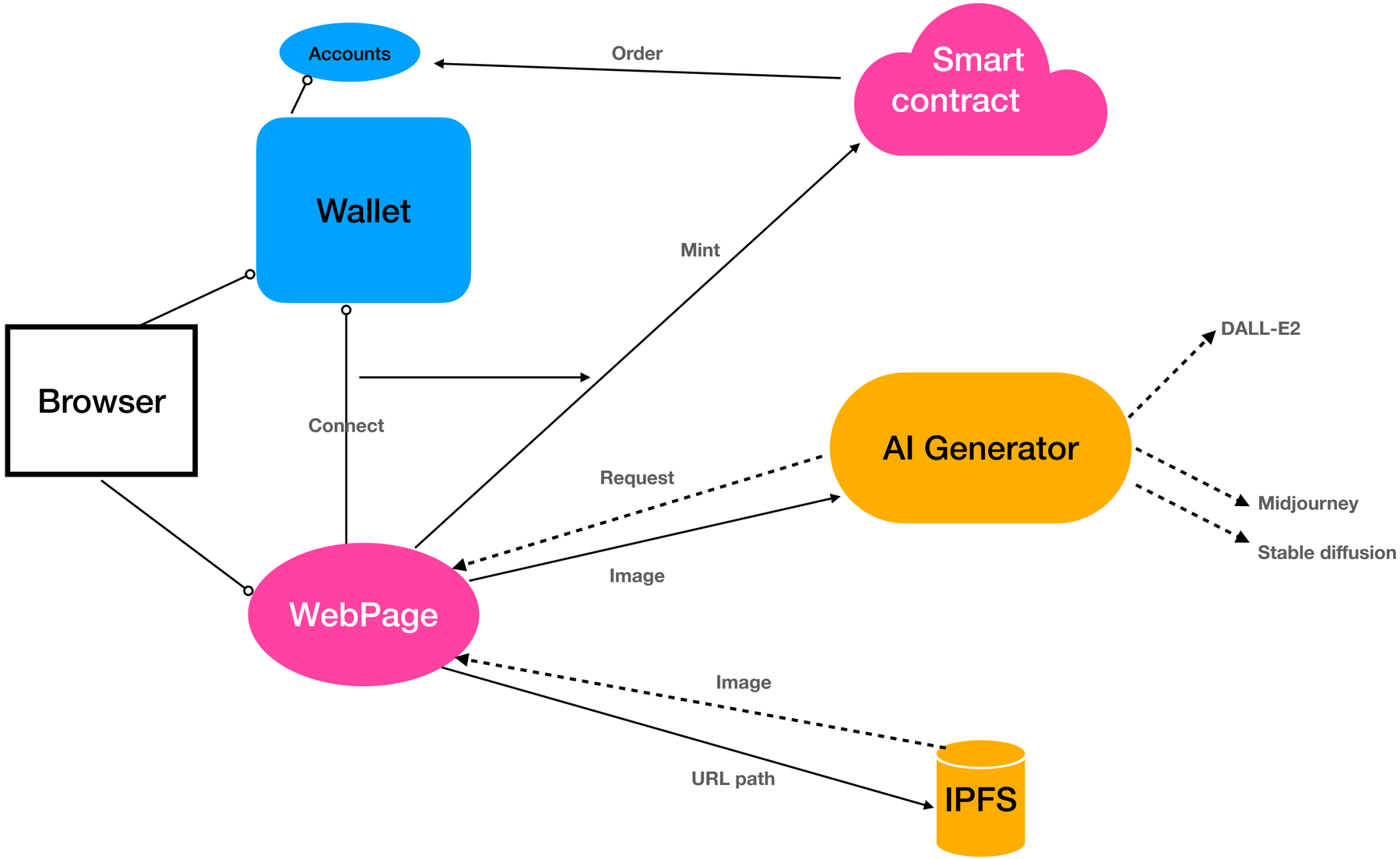
Stable Diffusion



To summarise …

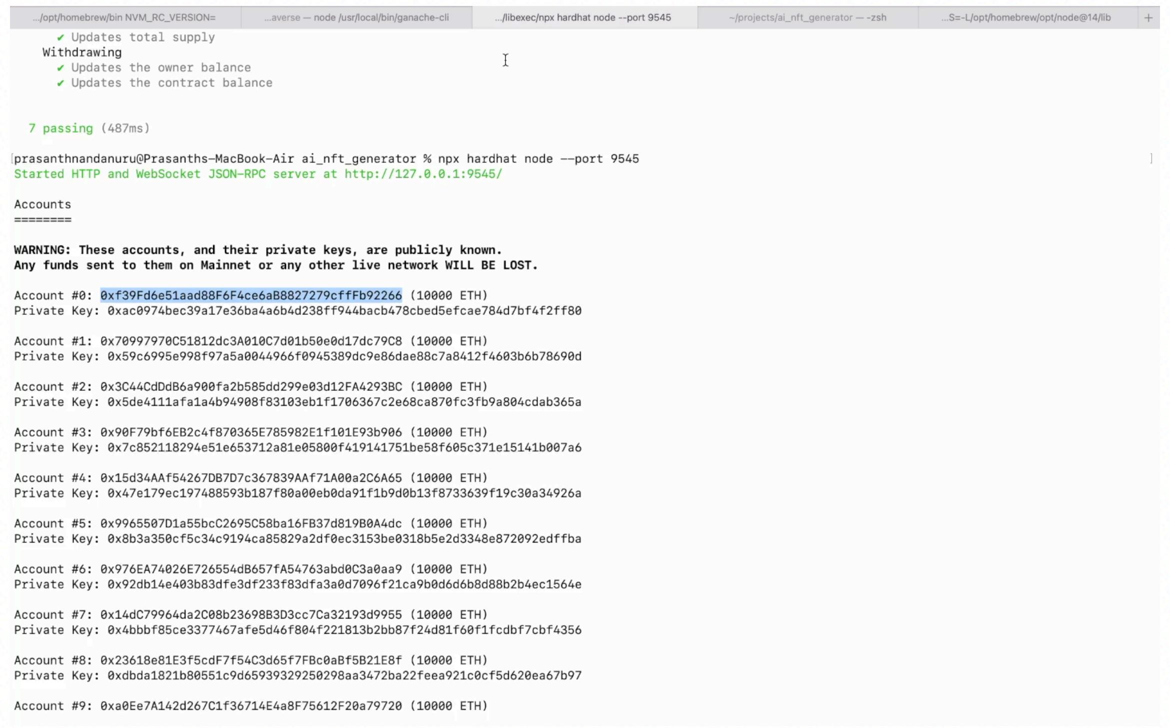
	VAE	Flow	GAN	Diffusion
Pros	Fast Sampling rate. Diverse sample generation	Fast Sampling rate. Diverse sample generation	Fast Sampling rate. High sample generation quality.	High sample generation quality. Diverse sample generation
Cons	Low sample generation quality	Need specialized architecture, low sample generation quality	Unstable training, low sample generation diversity (Mode Collapse)	Low sampling rate

NFT Generator - Stable diffusion Model



NFT Generator - Stable diffusion model

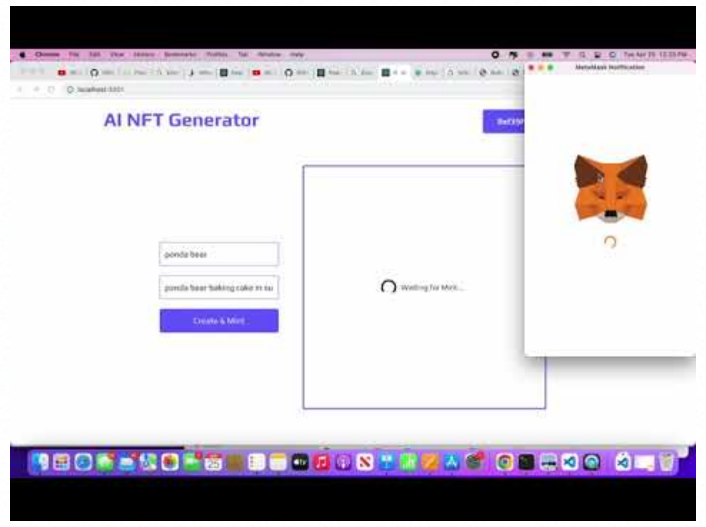
0xf39F...2266



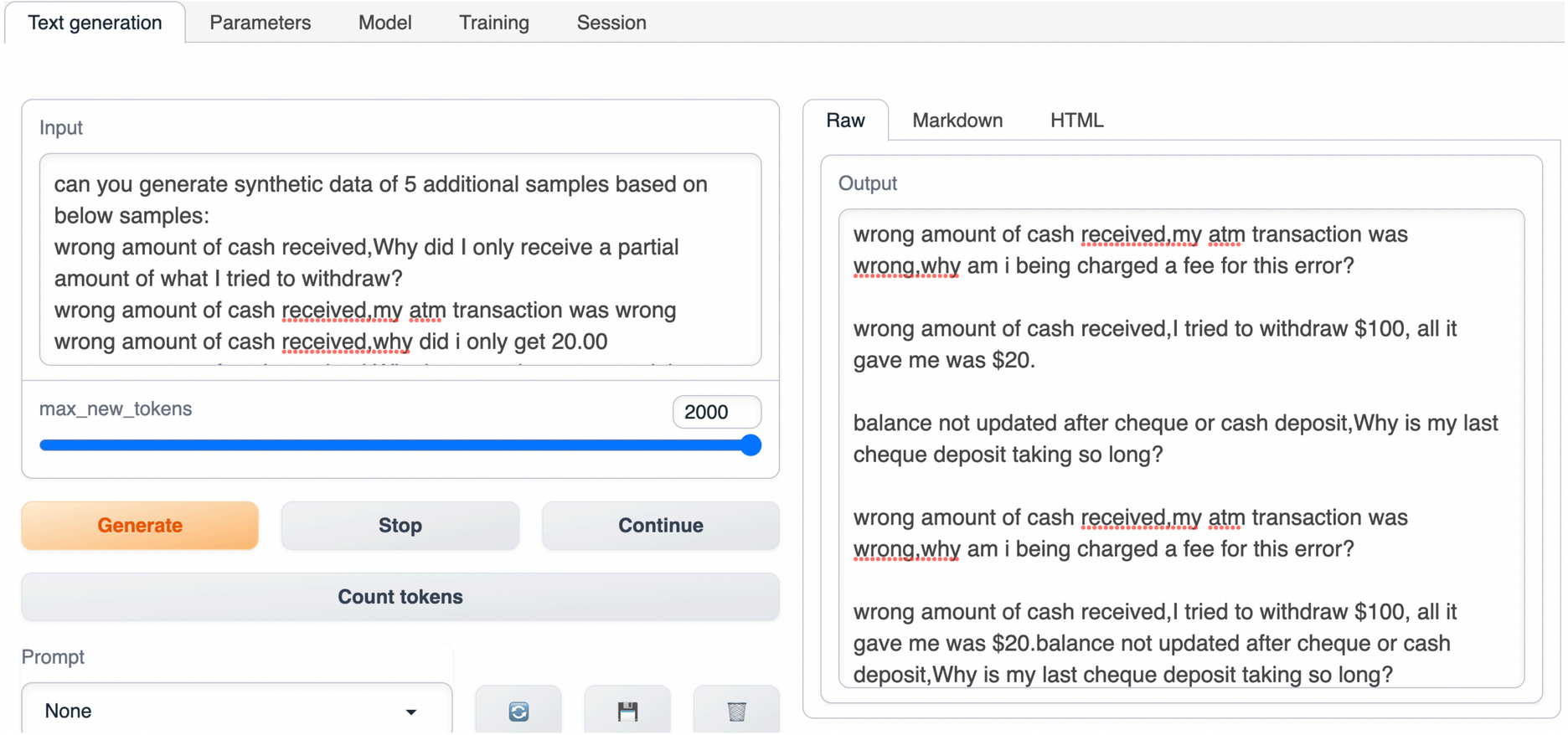
Teddy Bear

Teddy bear with sparkling wi

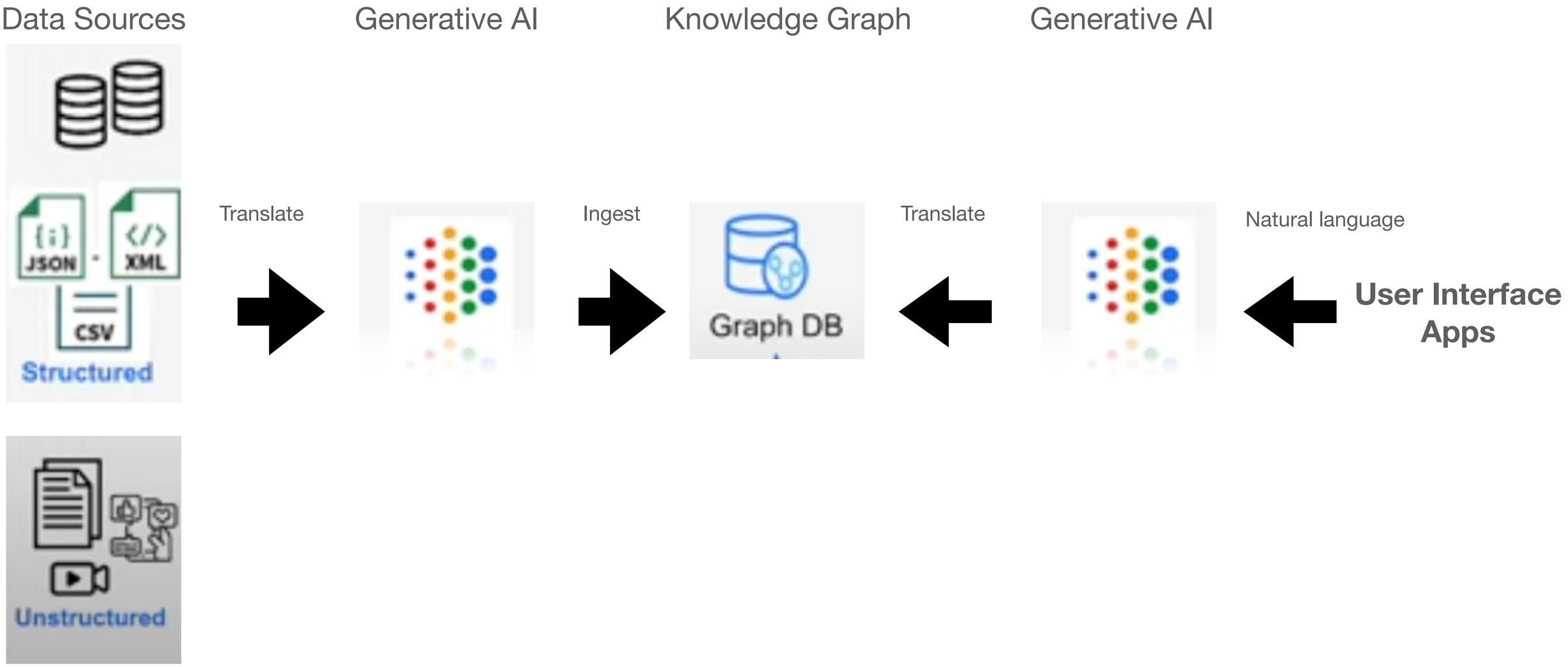
Create & Mint



Generative Synthetic Text using LLM and Prompt engineering



Knowledge Graph and Generative AI



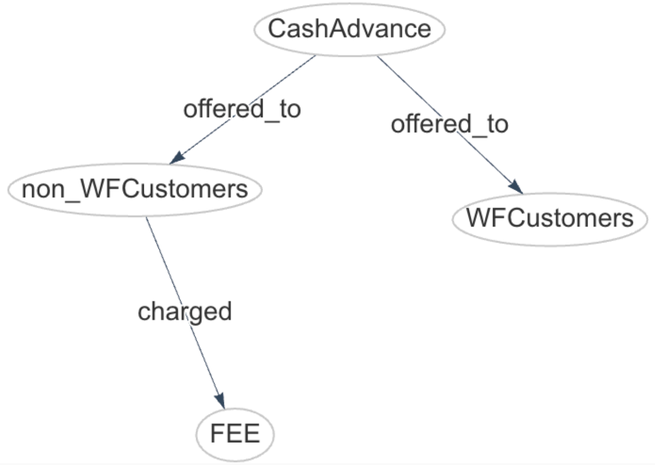
Graph AI

Prompt:

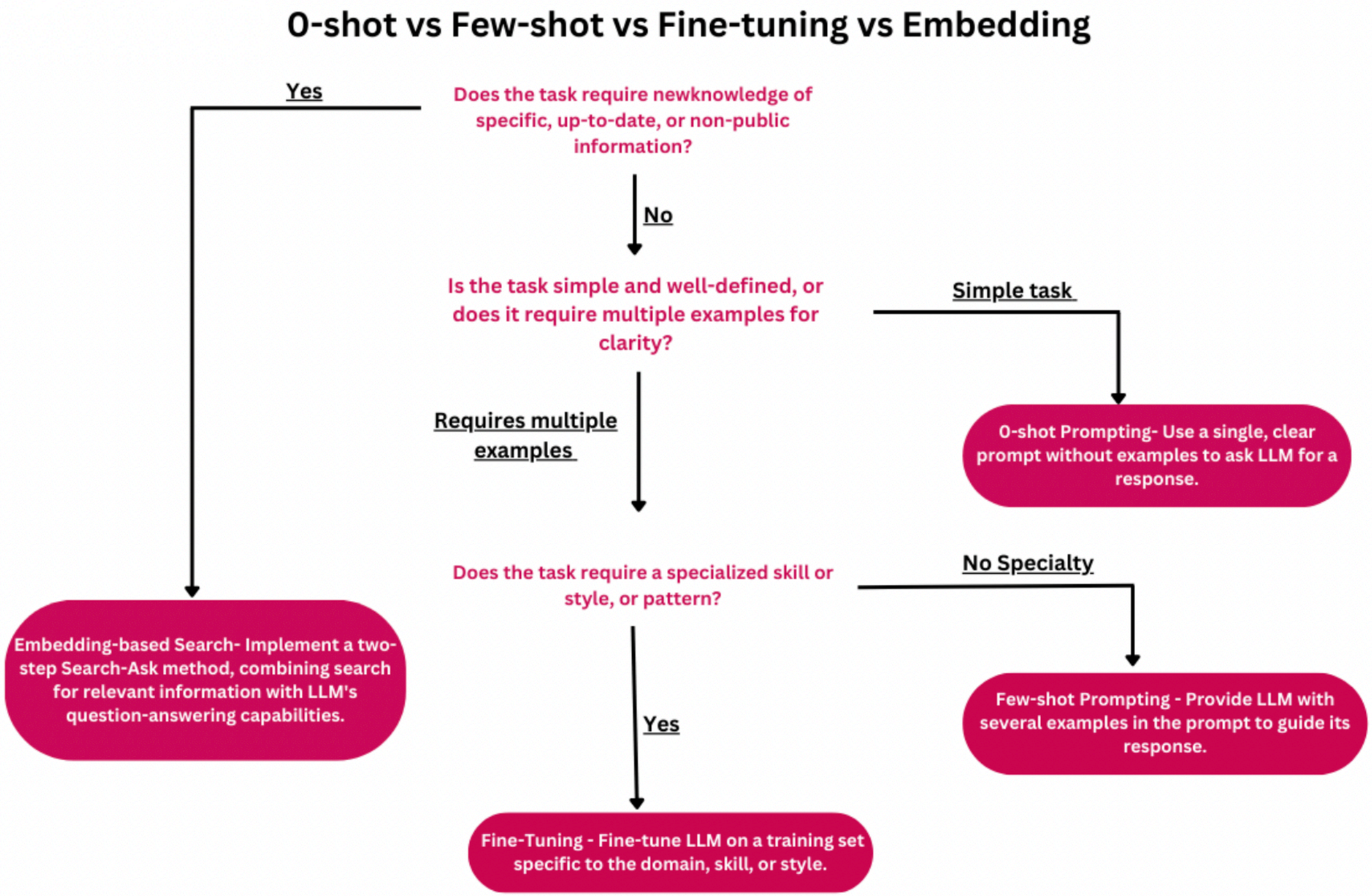
CashAdvance is offered to WFCustomers and non WFCustomers who are charged FEE.

Generate | Clear

Graph Response:

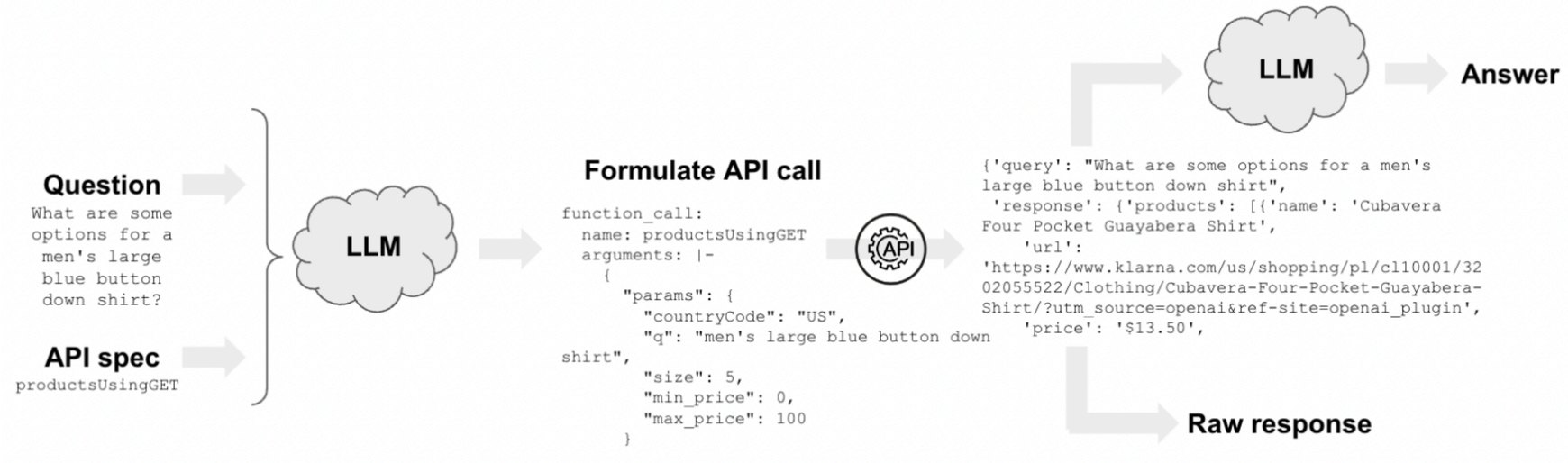


Chosing right LLM Strategy for text generation



AI Guardrails -Langchain or Semantic Kernel Augmentation

Langchain function to integrate with GuadRails Library to meet compliance and regulations PII/GDPR/CCPA



Nemo guardrails: AI with action on response

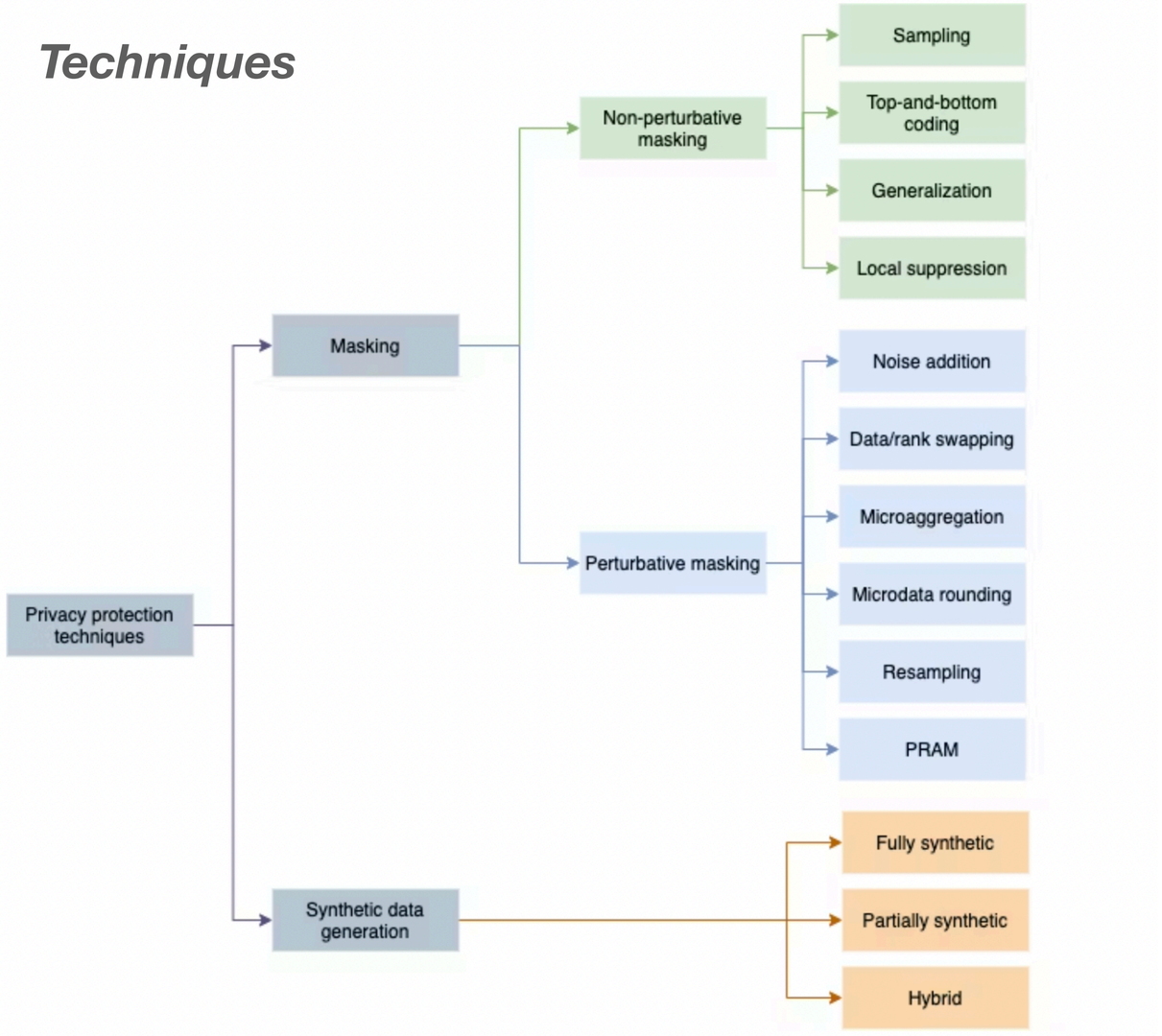
await rails.generate_async(prompt =“Tokenise PAN ################“)

```
async def func (inputs:str)
  #Do PAN tokenization logic
  panStr=pantokenise(str)
  return panStr
```

rails.register_action(action=func, name=“response”)

Privacy

Techniques



Models

K-Anonymity
K-Map
l-Diversity
t-CloseNess
Differential Privacy

Tools

ARX
Amnesia
Delphix
Anominatron

Security :OWASP Top 10

LLM01

Prompt Injection

This manipulates a large language model (LLM) through crafty inputs, causing unintended actions by the LLM. Direct injections overwrite system prompts, while indirect ones manipulate inputs from external sources.

LLM02

Insecure Output Handling

This vulnerability occurs when an LLM output is accepted without scrutiny, exposing backend systems. Misuse may lead to severe consequences like XSS, CSRF, SSRF, privilege escalation, or remote code execution.

LLM03

Training Data Poisoning

Training data poisoning refers to manipulating the data or fine-tuning process to introduce vulnerabilities, backdoors or biases that could compromise the model's security, effectiveness or ethical behavior.

LLM04

Model Denial of Service

Attackers cause resource-heavy operations on LLMs, leading to service degradation or high costs. The vulnerability is magnified due to the resource-intensive nature of LLMs and unpredictability of user inputs.

LLM05

Supply Chain Vulnerabilities

LLM application lifecycle can be compromised by vulnerable components or services, leading to security attacks. Using third-party datasets, pre- trained models, and plugins add vulnerabilities.

LLM06

Sensitive Information Disclosure

LLMs may inadvertently reveal confidential data in its responses, leading to unauthorized data access, privacy violations, and security breaches. Implement data sanitization and strict user policies to mitigate this.

LLM07

Insecure Plugin Design

LLM plugins can have insecure inputs and insufficient access control due to lack of application control. Attackers can exploit these vulnerabilities, resulting in severe consequences like remote code execution.

LLM08

Excessive Agency

LLM-based systems may undertake actions leading to unintended consequences. The issue arises from excessive functionality, permissions, or autonomy granted to the LLM-based systems.

LLM09

Overreliance

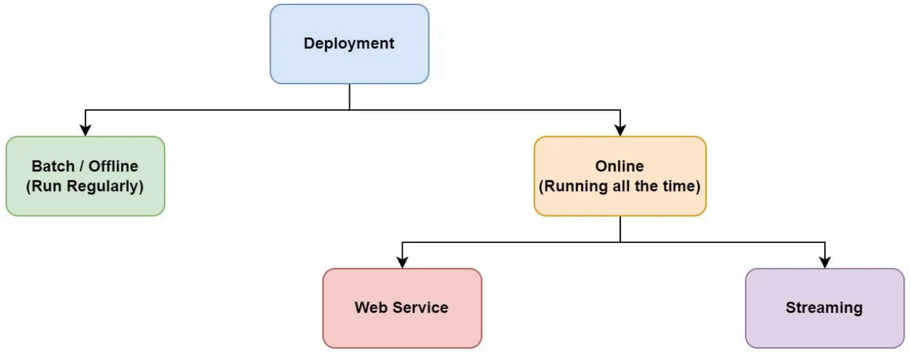
Systems or people overly depending on LLMs without oversight may face misinformation, miscommunication, legal issues, and security vulnerabilities due to incorrect or inappropriate content generated by LLMs.

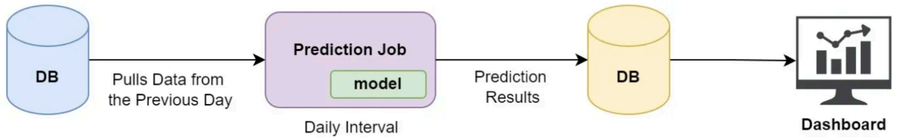
LLM10

Model Theft

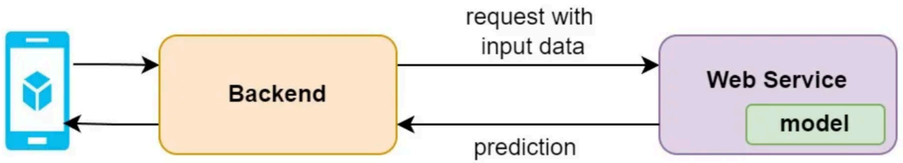
This involves unauthorized access, copying, or exfiltration of proprietary LLM models. The impact includes economic losses, compromised competitive advantage, and potential access to sensitive information.

Deployment approaches

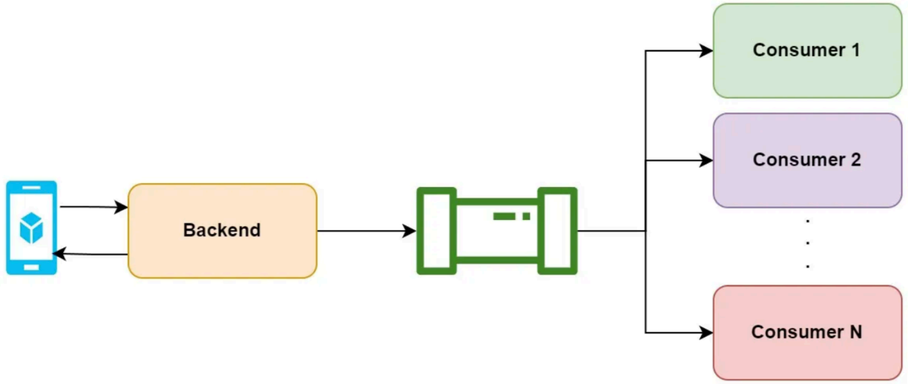




Batch



Real time



Near Real time (Streaming)

Summary

- Time Series Problems require domain knowledge
- Selection of models is dependent on business goals
  Training time vs Performance vs Time to Market
- Synthetic data generation is not just for Testing . Use for Remix and Simulation of business models
- Text and Image Generation using Generative AI solutions are gaining lot of traction and competitive advantage versus Discriminative AI
- Privacy , XAI are not good to have but must to account for for any AI solution.

References

https://papers.nips.cc/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf
https://github.com/openai/CLIP
https://openai.com/research/clip
https://github.com/microsoft/semantic-kernel
https://docs.langchain.com/docs/use-cases/apis
https://github.com/NVIDIA/NeMo-Guardrails

THANK YOU