1. GPT (Generative Pre-trained Transformer)

What It Is:

GPT is a **language model** developed by OpenAI that generates human-like text. It's based on the **Transformer architecture** and trained on massive datasets.

How It Works:

- **Pre-training**: GPT learns by predicting the next word in a sentence across billions of examples.
- **Fine-tuning**: It's later adapted to specific tasks (e.g., summarization, translation).
- **Autoregressive**: It generates text one token at a time, using previous tokens as context.

Key Features:

- Understands context and nuance.
- Can write essays, poems, code, emails, etc.
- Powers ChatGPT and other conversational agents.

Core Architecture: Transformer Decoder

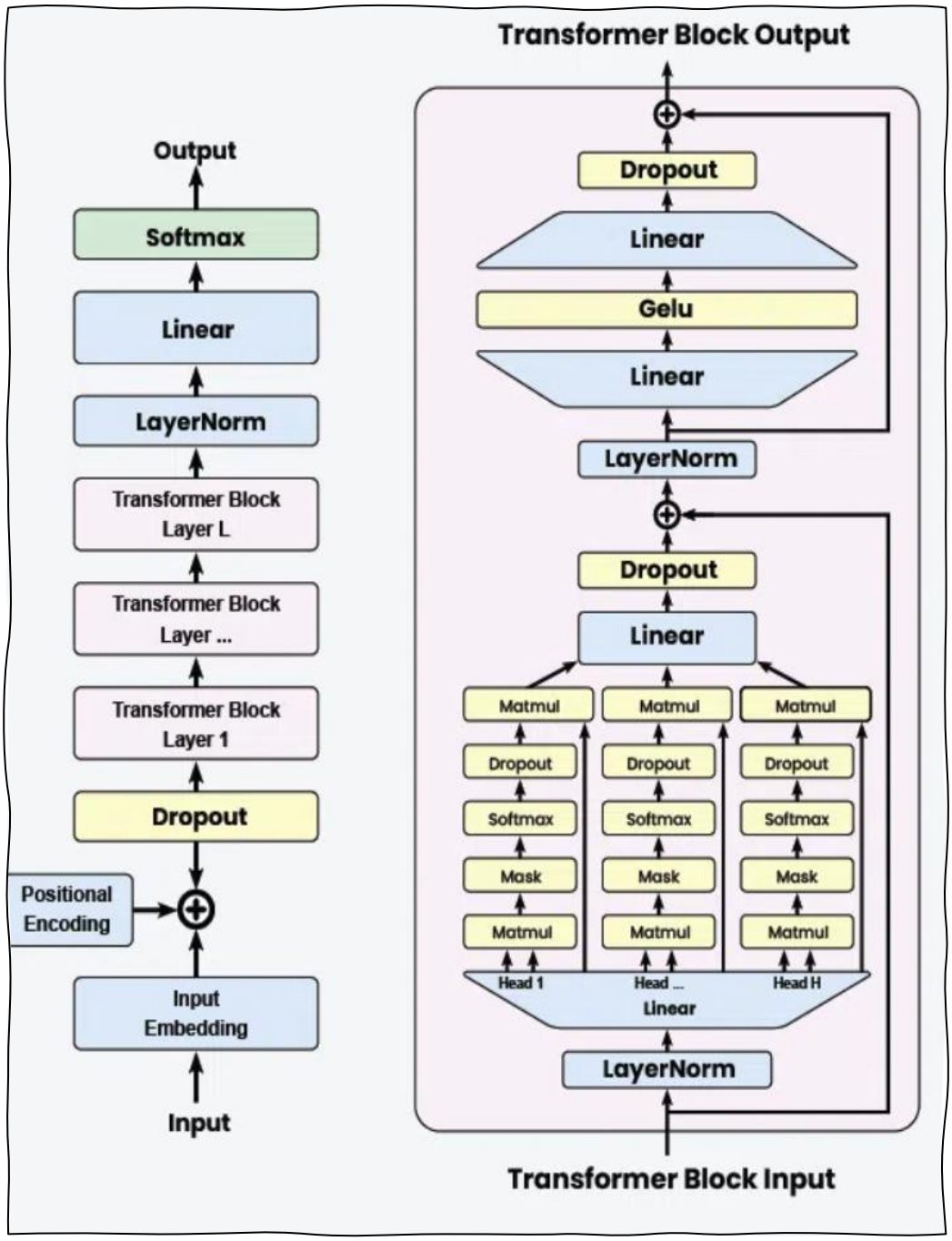
GPT is built entirely on the **decoder part** of the Transformer architecture.

Key Components:

1. **Input Embedding**: Converts words (tokens) into vectors.
2. **Positional Encoding**: Adds position info to each token (since Transformers don't inherently understand order).
3. **Stacked Decoder Blocks**:
    - **Masked Multi-Head Self-Attention**: Each token attends only to previous tokens (causal masking).
    - **Feedforward Neural Network**: Applies transformations to each token.
    - **Layer Normalization & Residual Connections**: Stabilizes training and preserves gradients.
4. **Output Layer**: Predicts the next token using a softmax over vocabulary.

Training:

- **Objective**: Minimize the difference between predicted and actual next token.
- **Data**: Trained on massive corpora (books, websites, code, etc.).



2. DALL·E

What It Is:

DALL·E is a **text-to-image generation model** by OpenAI. It creates images from natural language prompts.

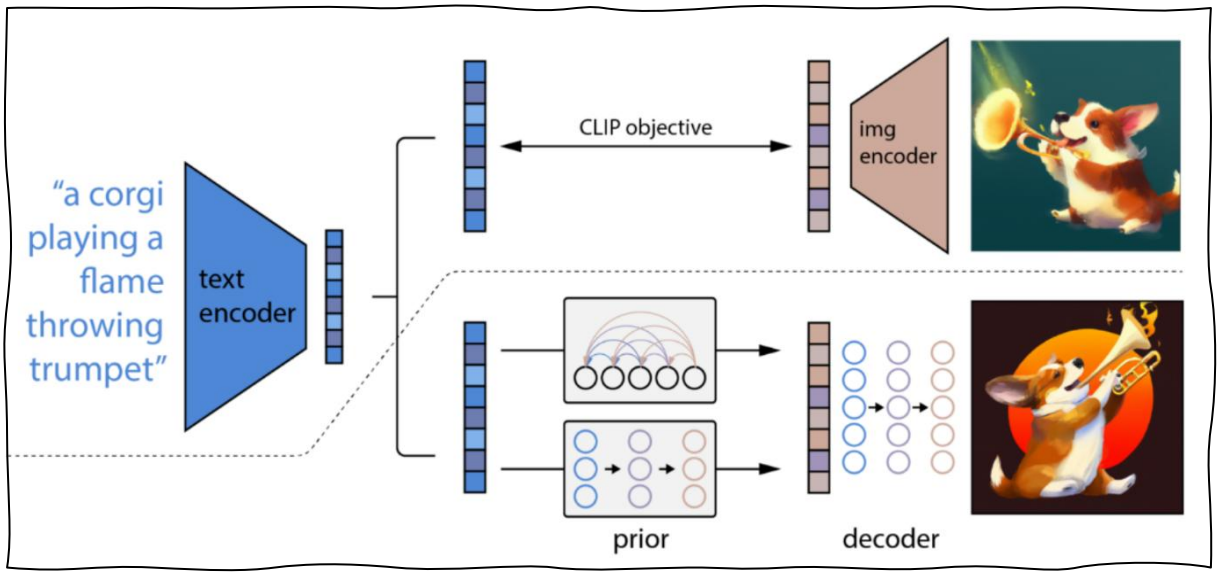
How It Works:

- Combines **Transformer architecture** with **diffusion models** (in newer versions).
- Learns associations between text and visual elements.
- Generates images pixel-by-pixel or via latent representations.

Key Features:

- Can generate surreal, realistic, or artistic images.
- Understands complex prompts like "a cat wearing a space suit riding a bike."

Architecture:



Components:

1. **CLIP (Contrastive Language-Image Pretraining)**:
    - Trained to align text and image embeddings.
    - Helps understand the prompt and guide image generation.
2. **Prior Model**:
    - Converts text embeddings into image latent space.
    - Uses a **Transformer** or **Diffusion model**.

3. **Decoder (Image Generator)**:
    - Uses **Diffusion** to generate images from latent representations.

Gradually denoises a random image to match the prompt

3. Codex

What It Is:

Codex is a specialized version of GPT trained on **code**. It powers tools like **GitHub Copilot**.

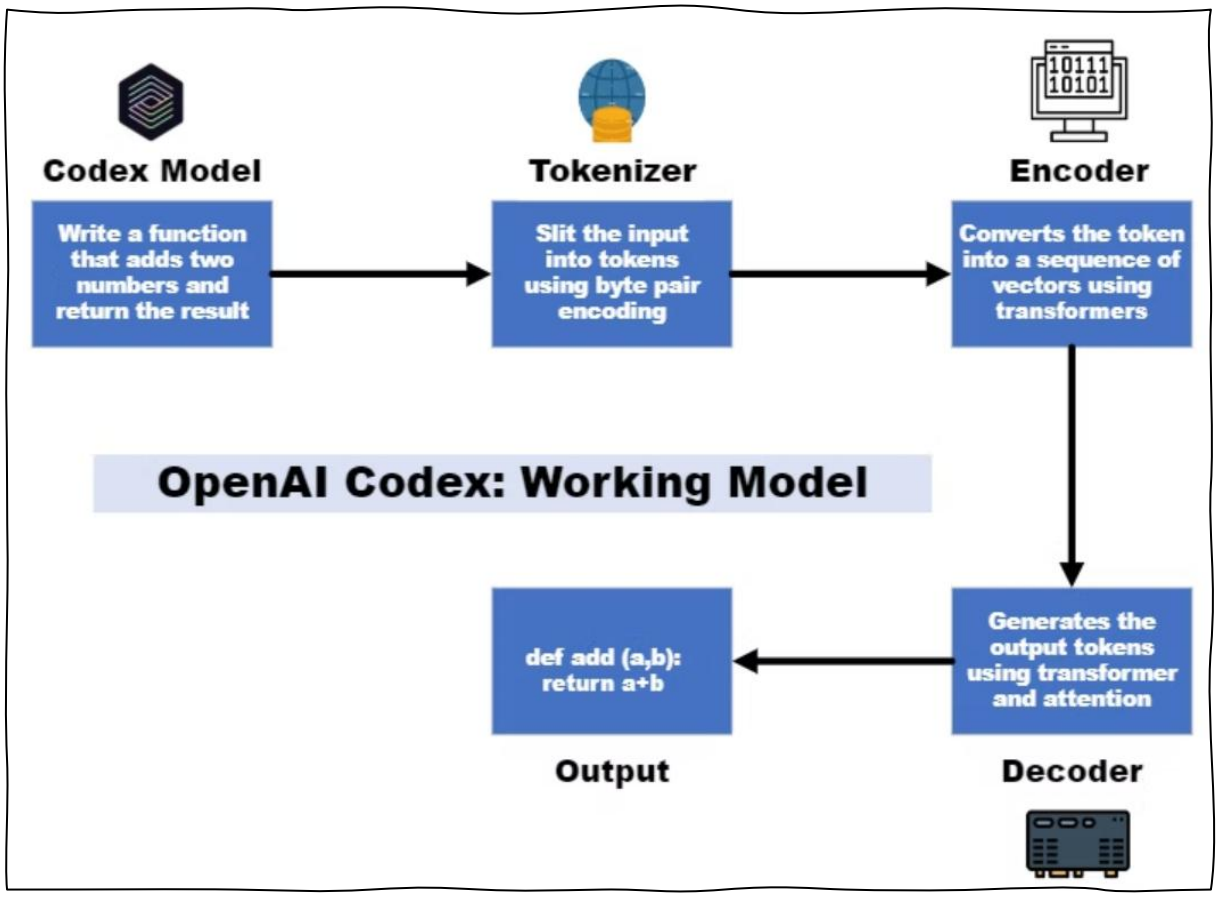
How It Works:

- Trained on billions of lines of code from public repositories.
- Understands programming languages like Python, JavaScript, C++, etc.
- Can generate, explain, and debug code.

Key Features:

- Converts natural language to code.
- Assists in writing functions, scripts, and even full applications.
- Supports multiple programming languages.

Architecture:



Codex is a **fine-tuned version of GPT**, specialized for programming.

Core Architecture: Same as GPT

Differences:

- **Training Data**: Includes billions of lines of code from GitHub and other sources.
- **Tokenization**: Handles code-specific tokens (e.g., brackets, keywords).
- **Capabilities**: Understands syntax, logic, and even documentation.

Use Cases:

- Code completion
- Natural language to code translation
- Explaining code.

4. Stable Diffusion

What It Is:

Stable Diffusion is a **text-to-image model** that uses **diffusion techniques** to generate high-quality images.

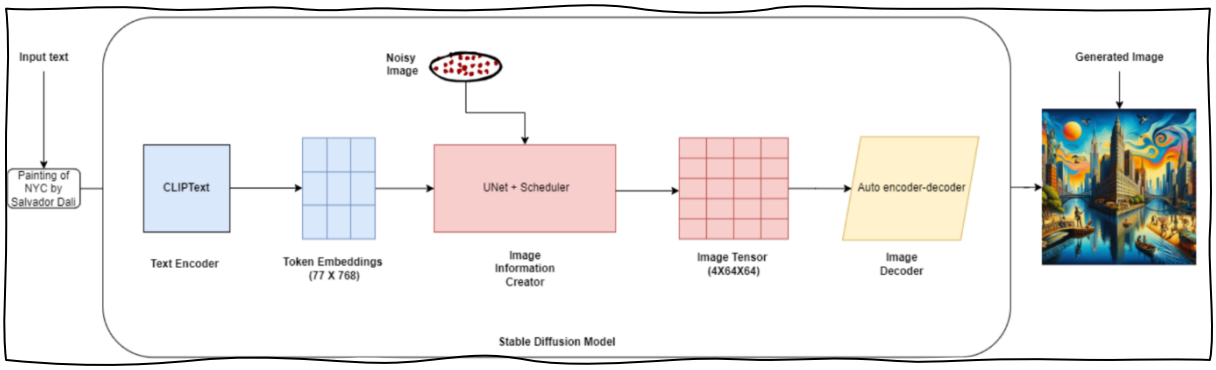
How It Works:

- Starts with random noise and gradually "denoises" it to form an image.
- Guided by a text prompt using **CLIP embeddings**.
- Operates in **latent space** for efficiency.

Key Features:

- Open-source and highly customizable.
- Can generate photorealistic or stylized images.
- Supports inpainting, outpainting, and image-to-image generation.

Architecture:



Components:

1. **Variational Autoencoder (VAE)**:
    - Compresses images into a latent space.
    - Reduces computational cost.
2. **U-Net (Denoising Network)**:
    - Takes noisy latent image and denoises it step-by-step.
    - Guided by text embeddings.
3. **CLIP Text Encoder**:
    - Converts text prompts into embeddings.

- Guides the denoising process to match the prompt.

4. **Scheduler**:
    - Controls the noise levels during generation.
    - Determines how many steps to take.

Core Technologies Explained

A. Transformers

What They Are:

Transformers are neural network architectures designed for **sequence modeling** — especially in NLP.

Key Concepts:

- **Self-attention**: Each word attends to every other word in the sequence.
- **Positional encoding**: Adds order information to input tokens.
- **Encoder-decoder**: GPT uses decoder-only; BERT uses encoder-only.

Applications:

- Language models (GPT, BERT)
- Text-to-image (DALL·E)
- Code generation (Codex)

B. GANs (Generative Adversarial Networks)

What They Are:

GANs consist of two neural networks:

- **Generator**: Creates fake data.
- **Discriminator**: Evaluates whether data is real or fake.

How They Work:

- The generator tries to fool the discriminator.
- The discriminator gets better at spotting fakes.
- This adversarial process improves both.

Applications:

- Image generation
- Deepfakes
- Style transfer

C. Diffusion Models

What They Are:

Diffusion models generate data by **reversing a noise process**.

How They Work:

- Start with pure noise.
- Gradually remove noise using a trained denoising model.
- The final output is a coherent image or data sample.

Key Models:

- **DDPM (Denoising Diffusion Probabilistic Models)**
- **Latent Diffusion Models (used in Stable Diffusion)**

Applications:

- Image generation (Stable Diffusion)
- Audio synthesis
- Video generation

Foundational Technologies

A. Transformer Architecture

Invented by: Vaswani et al. (2017)

Key Concepts:

- **Self-Attention**: Each token looks at all others to understand context.
- **Multi-Head Attention**: Multiple attention mechanisms run in parallel.
- **Positional Encoding**: Adds sequence order info.
- **Feedforward Layers**: Non-linear transformations.
- **Residual Connections**: Prevent vanishing gradients.
- **Layer Normalization**: Stabilizes training.

Variants:

- **Encoder-only**: BERT
- **Decoder-only**: GPT
- **Encoder-Decoder**: T5, BART

B. GANs (Generative Adversarial Networks)

Invented by: Ian Goodfellow (2014)

Architecture:

- **Generator**: Creates fake data.
- **Discriminator**: Judges real vs fake.
- **Training Loop**: Generator tries to fool discriminator; discriminator improves at spotting fakes.

Challenges:

- Training instability
- Mode collapse (generator produces limited variety)

Use Cases:

- Image synthesis

- Super-resolution
- Style transfer

C. Diffusion Models

Concept:

- Start with noise → gradually remove noise → generate data.

Architecture:

- **Forward Process**: Adds noise to data over time.
- **Reverse Process**: Learns to remove noise step-by-step.
- **U-Net**: Core denoising network.
- **Scheduler**: Controls noise levels and steps.

Latent Diffusion:

- Operates in compressed latent space (via VAE).
- Much faster and memory-efficient.