Deep Neural Networks Project - 2025

Stratos Kakalis, ID: 7115152400039
Fotis Vorloou, ID: 7115152400002

January 2025

1 Introduction to Scientific Figure Synthesis SciFS

Scientific figures play a crucial role in transforming complex data and abstract concepts into intuitive visual representations that are not easily conveyed through text alone. They serve as essential tools in research, enabling clearer communication of findings and facilitating the understanding of intricate relationships between data points. However, their creation remains a highly labor-intensive and skill-dependent process, requiring expertise in tools such as Adobe Illustrator, Matplotlib, or LaTeX code.

This reliance on manual design methods presents a significant bottleneck, limiting rapid iteration and reducing the flexibility to adapt figures for different target audiences. The time and effort required to produce high-quality scientific visualizations often hinder the efficiency of rapid research prototyping and collaboration.

In this project, we explore the potential of modern generative AI technologies for automating the synthesis of scientific figures based on textual descriptions (e.g., captions). By leveraging image and code generating techniques, we aim to synthesize structured, high-quality visuals that preserve the accuracy and clarity required for scientific communication. Through a series of experiments, we evaluate multiple approaches to automated figure generation, analyze their effectiveness in producing meaningful and interpretable figures, and document key challenges such as the inability to easily fine-tune image-generating systems to generate scientific figures with meaningful structure.

2 Data Preparation

2.1 Dataset with Scientific Figures

For this study, we leveraged the **ACL-FIG** dataset, a well-curated collection of scientific figures extracted from 890 research papers in the *ACL Anthology*. The dataset comprises **1,758 figures**, each stored in PNG format, and is categorized into **19 distinct figure types**, covering a wide range of scientific visualizations commonly used in computational linguistics and machine learning research. The dataset provides essential metadata, including figure captions, inline references, and paper identifiers, making it an ideal resource for training generative AI models focused on scientific figure synthesis.

2.1.1 Categories in the ACL-FIG Dataset

The dataset contains figures representing diverse scientific visualization types, including:

- **Algorithms**: Flowcharts and structured algorithmic representations.
- **Architecture/Pipeline Diagrams**: Visual descriptions of model architectures and computational pipelines.
- **Bar Charts**: Categorical data visualizations using bar heights to indicate values.
- **Box Plots**: Statistical data summaries showing distributions, medians, and outliers.
- **Confusion Matrices**: Evaluation metrics for classification model performance.
- **Graphs**: Graph structures, node-based models, and relational diagrams.
- **Line Charts**: Trend analysis over time with continuous data.
- **Maps**: Geographical and spatial visualizations.
- **Natural Images**: Photographic images used within research.
- **Neural Networks**: Diagrams representing artificial neural networks and their layers.
- **NLP Rules/Grammar**: Formal grammar representations in natural language processing research.
- **Pie Charts**: Circular visualizations representing proportions.
- **Scatter Plots**: Distribution and correlation of data points.
- **Screenshots**: Extracted screenshots from software interfaces.

- **Tables**: Structured tabular data for numerical comparisons.
- **Trees**: Hierarchical structures and decision trees.
- **Pareto Charts**: Ranked bar charts combining frequency distributions.
- **Venn Diagrams**: Set relationships and logical groupings.
- **Word Clouds**: Frequency-based text visualizations.

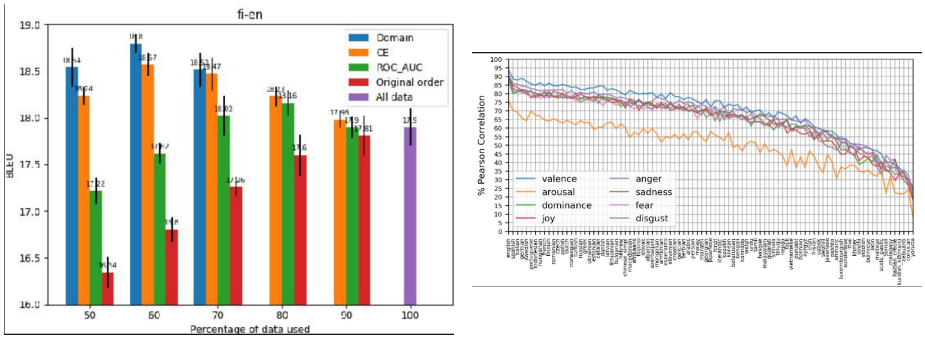

Figure 1: Examples of charts from the ACL-FIG dataset. The left image depicts a bar chart, while the right image represents a line graph, illustrating different types of visualizations available in the dataset.

2.2 How the Dataset Was Used in Training

To effectively train our models, we split the dataset into **training (80%)**, **validation (10%)**, and **test (10%)** sets. This split ensured a balanced evaluation, preventing the models from overfitting to specific figure types while allowing for generalization across unseen samples.

Different generative models were trained on this dataset, each employing distinct architectures and training strategies. Some models utilized an **autoencoder-based approach**, where figures were encoded into a latent space and then reconstructed, while others explored **adversarial training methods** or **diffusion-based techniques**. The training objective varied based on the model type but generally aimed to minimize reconstruction loss, ensuring that generated figures retained structural accuracy.

Additionally, **data augmentation** techniques such as *random cropping, flipping, and resizing* were applied to improve robustness and prevent overfitting. These transformations helped introduce variations in the training data, ensuring that the models learned more general representations rather than memorizing specific figure layouts.

2.3 Comparison with Other Datasets

While ACL-FIG served as a valuable dataset for training our models, its relatively small size (1,758 images) limited the diversity of generated figures. Other datasets, such as **PubLayNet** and **arXiv Figures**, offer significantly larger repositories of scientific images, making them potential candidates for future extensions of this work. However, ACL-FIG was chosen due to its **well-structured metadata** and **fine-grained categorization**, which are essential for learning figure semantics.

In comparison to datasets that focus on **document layouts** or **general diagrams**, ACL-FIG is uniquely tailored to **scientific visualizations**, making it a better fit for our research objectives. However, one major limitation is the **skewed category distribution**, where some figure types (e.g., bar charts and tables) are overrepresented while others (e.g., word clouds and Pareto charts) are underrepresented. This imbalance may have affected the models' ability to generalize equally across all figure types.

For future work, integrating ACL-FIG with larger datasets could provide better model robustness since the task at hand is very complex and a large training dataset will likely prove essential.

3 Figure Synthesis based on Image-Generation Techniques

Automated figure synthesis has become an essential task in various scientific and technical domains, enabling the efficient generation of structured visualizations such as charts, schematics, and diagrams. Traditional methods for figure creation rely on predefined templates and rule-based systems, which lack flexibility and require manual intervention. Recent advancements in deep generative models have introduced data-driven approaches capable of learning complex visual representations, offering a more adaptive and scalable solution for figure generation. In this section, we explore three major deep learning-based techniques for figure synthesis: Generative Adversarial Networks (GANs), Latent Diffusion Models (LDMs), and Vision Transformers (ViTs). Each method is analyzed in terms of its theoretical framework, training mechanisms, and empirical performance in structured image generation. Through comparative analysis, we examine their effectiveness in capturing figure semantics, enforcing structural coherence, and maintaining high-quality visual fidelity.

3.1 GAN-based Figure Synthesis

Generative Adversarial Networks (GANs) are a class of deep learning models designed to generate synthetic data that resembles real-world samples. They consist of two neural networks, the **Generator (G)** and the **Discriminator (D)**, which compete against each other in a zero-sum game:

- **Generator (G)**: Learns to create synthetic samples from random noise.
- **Discriminator (D)**: Learns to distinguish real samples from fake ones produced by the generator.

Through adversarial training, the generator improves its ability to generate realistic images, while the discriminator becomes better at detecting fake ones. This dynamic continues until the generator produces samples that are indistinguishable from real data.

Generative Adversarial Networks (GANs) have been widely explored in the field of image synthesis due to their ability to model complex data distributions through adversarial training. Conditional GANs (cGANs) extend this framework by introducing label-conditioning, allowing controlled generation of images based on specified classes. This capability makes them a potential solution for scientific figure synthesis, where different visualization types (e.g., bar charts, line graphs, and schematics) must be generated with label-based guidance.

However, unlike natural images, scientific figures require precise geometric alignment, text readability, and adherence to structured layouts. These constraints pose significant challenges to cGANs, which primarily operate through holistic feature learning rather than explicitly modeling structured components. To evaluate the viability of cGANs for scientific figure generation, we conducted

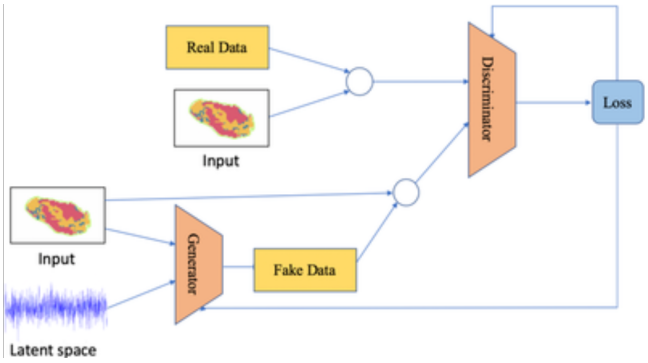

Figure 2: Visualization of conditional GAN architecture.

multiple experiments and analyzed the resulting outputs through qualitative and quantitative assessments.

3.1.1 Implementation of the GAN Model

In this implementation, the generator takes both **random noise** and a **label** as inputs to generate images of a specific category. Similarly, the discriminator learns to distinguish real and fake images while taking the label into account.

The implemented GAN consists of two main components: The **Generator** network is responsible for transforming a random noise vector into a structured image that resembles a scientific figure. The architecture follows a **fully connected layer followed by transposed convolutional layers** to upsample the noise into an image.

Key Components of the Generator:

- **Label Embedding:**
  - Labels are embedded using `nn.Embedding(num_classes, num_classes)`, which converts class indices into learnable vectors.
  - This allows the generator to condition the image generation process on class-specific information.
- **Linear Projection and Upsampling:**
  - The noise and label embedding are concatenated into a **single latent vector**.
  - This vector is projected to a higher-dimensional space and reshaped into a **small feature map**.
  - **Upsampling layers** (via `nn.Upsample`) progressively increase the resolution of the feature map.

- **Convolutional Refinement:**
  - The upsampled feature maps pass through **several convolutional layers** (`nn.Conv2d`).
  - Each convolution is followed by **batch normalization** (`nn.BatchNorm2d`) to stabilize training and **Leaky ReLU activation** to introduce non-linearity.
  - A final **tanh activation** (`nn.Tanh()`) ensures that pixel values are normalized between $[-1, 1]$.

The **Discriminator** network is a binary classifier that determines whether an image is **real** (from the dataset) or **fake** (generated by G). It also incorporates label conditioning, ensuring that the model considers class labels when making decisions.

Key Components of the Discriminator:

- **Label Embedding:**
  - Similar to the generator, labels are embedded using `nn.Embedding(num_classes, num_classes)`.
  - This allows the model to condition its decision-making process on class-specific information.
- **Flattening and Concatenation:**
  - The input image is **flattened into a vector** using `.view(img.size(0), -1)`, converting it from (C, H, W) to $(batch\_size, flattened\_dim)$.
  - The label embedding is concatenated with the image vector to form a **single input representation**.
- **Fully Connected Classifier:**
  - The concatenated vector is passed through **several fully connected** (`nn.Linear`) **layers**.
  - **Leaky ReLU activations** (`nn.LeakyReLU(0.2)`) help prevent dead neurons.
  - A final **sigmoid activation** (`nn.Sigmoid()`) produces a probability score between $[0, 1]$, indicating the likelihood that the image is real.

3.1.2 Experiments on GAN-Based Figure Generation

To investigate the effectiveness of cGANs in scientific figure synthesis, we trained a model on a dataset of extracted research figures. Our analysis focused on three key objectives:

- **Label Consistency**: Determining whether generated figures accurately correspond to their assigned categories.

- **Mode Collapse**: Evaluating the diversity of generated figures across multiple noise inputs.
- **Visual Fidelity**: Assessing the structural integrity, coherence, and overall clarity of generated images.

Visual Evaluation of Generated Figures Upon completion of training, the generated images were visually inspected for structural consistency and meaningful representation of figures. The results revealed significant artifacts and distortions:

- **Blurry and Incomplete Figures**: Generated images frequently exhibited low clarity, with missing axes, misaligned gridlines, and distorted structural elements.
- **Noise-Dominated Outputs**: Many images appeared as random noise textures rather than recognizable scientific visualizations.
- **Lack of Semantic Alignment**: The figures often failed to resemble their corresponding categories, demonstrating weak conditional mapping between labels and visual output.
- **Mode Collapse**: The generator repeatedly produced near-identical outputs across different noise vectors, failing to capture a diverse range of figure structures.

A set of representative generated images is shown in Figure 4. These results confirm that the model fails to generate structured figures, producing visually incoherent patterns instead.

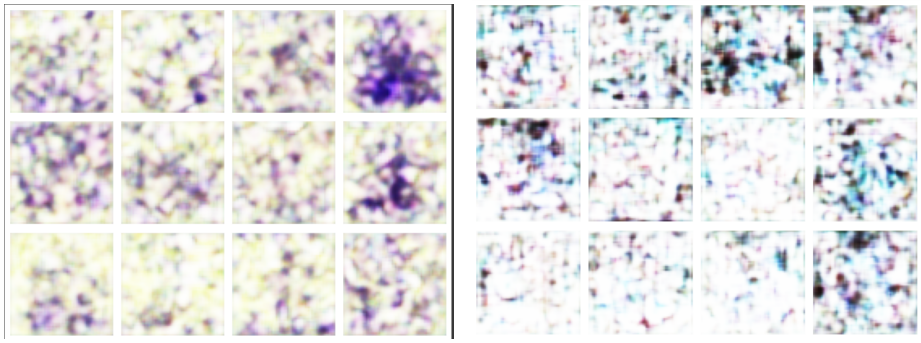

Figure 3: Example of generated figures from a trained cGAN model trying to depict a Neural Network chart (right) and a tree chart (left). The outputs primarily consist of noise-like artifacts, lacking meaningful figure structures.

Quantitative Performance Analysis To further evaluate the performance of the cGAN model, we computed the following quantitative metrics:

- **Inception Score (IS)**: Measures the realism and diversity of generated images by assessing classifier confidence. A low IS score suggests poor semantic quality and lack of category differentiation.
- **Fréchet Inception Distance (FID)**: Quantifies the distance between real and generated image distributions. High FID scores indicate weak structural similarity between generated and real figures.

Results and Observations The computed evaluation metrics further confirm the model's difficulties in synthesizing structured scientific figures:

- **The low IS score** ($\approx 1.47 \pm 0.10$) indicates that the generated images exhibit minimal diversity and fail to produce well-distinguished outputs across different figure categories. This suggests that the generator struggles with mode collapse, repeatedly generating similar unstructured outputs.
- **The high FID score** (≈ 315.0) highlights a substantial discrepancy between real and generated figures, reinforcing the conclusion that the synthesized images lack geometric coherence. The figures produced remain far from structurally accurate scientific visualizations, often appearing as noise or distorted shapes.
- **Loss analysis and latent space activations** reveal inconsistent training behavior, with the generator struggling to balance adversarial feedback. The discriminator's loss remains relatively stable, while the generator exhibits fluctuations, indicating weak conditional learning and difficulty in aligning latent representations with structured figure semantics.

Despite extended training (30 epochs), only marginal improvements in FID and IS scores were observed, suggesting that standard cGAN architectures are inherently ill-suited for structured figure synthesis. The findings indicate that GAN-based approaches struggle with enforcing spatial constraints and producing coherent, high-resolution figures. The observed failure cases suggest that the GAN models needed for the accurate depiction of scientific charts, are much more complex than what we can achieve on our own.

3.1.3 Failure Cases and Limitations

The failure of the cGAN model can be primarily attributed to two critical constraints: the **limited model capacity** and the **small dataset size**. These factors significantly impact the generator's ability to synthesize structured scientific figures, ultimately making it **impossible to achieve high-quality results with the current implementation**.

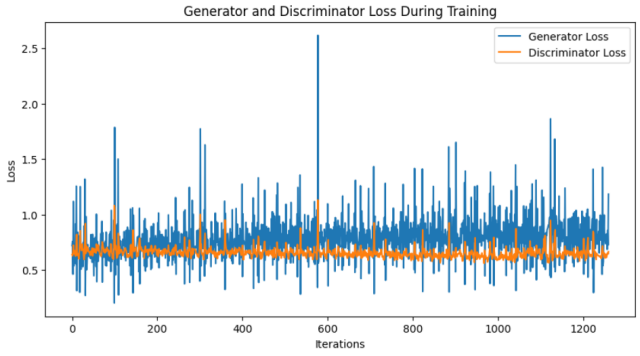

Figure 4: Plot showing the loss of the generator and the discriminator over 30 epochs of training on the ACL-FIG dataset.

Limited Model Capacity The current generator and discriminator architectures lack sufficient depth and complexity to model the intricate spatial relationships required for scientific figures. Unlike natural image synthesis, where GANs can exploit texture-based representations, structured figure generation requires the model to learn precise **geometric arrangements, text positioning, and well-defined graphical elements**. The observed failure cases, including **blurry outputs, missing figure components, and noisy artifacts**, suggest that the generator does not have the representational power needed to capture these structured relationships.

A deeper network with more convolutional layers, residual connections, and improved feature extraction would be necessary to enhance **spatial consistency** and **semantic accuracy**. However, increasing model complexity also demands significantly more training data and computational resources, which are currently insufficient in our implementation.

Small Dataset Size GANs require large, diverse datasets to learn a meaningful data distribution and avoid mode collapse. In our case, the dataset contains a **limited number of scientific figures**, leading to a restricted range of visual patterns the model can learn. With insufficient data, the generator fails to generalize across different figure types, resulting in:

- **Mode collapse**, where the generator converges to a **few dominant patterns**, repeatedly producing near-identical outputs.
- **Lack of label differentiation**, where the model **fails to associate input labels with distinct visual representations**, making conditional control ineffective.
- **Overfitting to a small subset of figures**, preventing the model from learning a diverse set of figure structures.

The dataset's small size also **hinders the discriminator's ability** to provide strong learning signals. With a limited number of real images to compare against, the discriminator may **overfit** to specific dataset patterns, reducing its effectiveness in guiding the generator toward more realistic outputs.

Impossibility of High-Quality Figure Generation in Our Implementation Given the constraints of **both the model architecture and dataset size**, it is not feasible to achieve high-quality scientific figures using our current cGAN implementation. The results show that:

- The generator struggles to create structured, coherent images due to its **limited capacity**.
- The dataset is too small for the model to **learn complex visual distributions**, leading to **low diversity and poor structural accuracy**.
- The training process is inherently unstable due to **adversarial imbalance**, exacerbated by the lack of sufficient data to refine the generator's outputs.

Even after **extended training (30 epochs)**, **only marginal improvements in IS and FID scores** were observed, reinforcing the fundamental limitations of the current approach. Achieving realistic figure synthesis would require a **larger dataset, a more expressive model architecture, and significantly more computational resources**. Without these improvements, cGANs remain unsuitable for structured figure generation in this setting.

3.2 Latent Diffusion Models (LDMs) for Figure Synthesis

3.2.1 Theory on LDMs

Diffusion models (DMs) revolutionized image generation by iteratively denoising random noise into coherent images through a learned noise scheduler. This scheduler controls the injection and removal of Gaussian noise across discrete timesteps during training, enabling the model to gradually reconstruct data distributions. We can see an example of such a model in Figure 5. While groundbreaking, DMs operate on pixel-space. Thus they suffer from high computational costs due to raw image dimensions being very large (e.g., $512\times512\times3$), making training and inference very expensive.

Latent Diffusion Models (LDMs) addressed this bottleneck by shifting computation to a compressed latent space learned by a variational autoencoder (VAE). The VAE's encoder compresses images into lower-dimensional latent representations, preserving semantic features while reducing memory and compute demands. During generation, LDMs denoise latent vectors, not pixels. Once generation is complete they decode the final latent vector into high-resolution images via the VAE's decoder. We can see this process, as opposed to the simpler DM in Figure 6. This efficiency makes LDMs the preferred tool to use

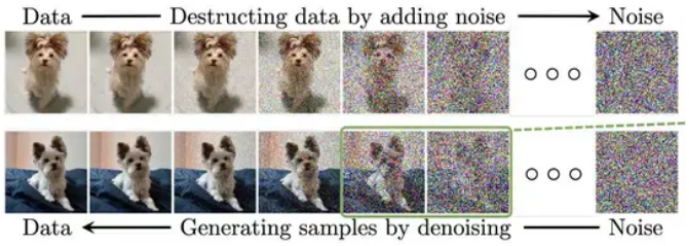

Figure 5: Standard Diffusion Process (training and generation).

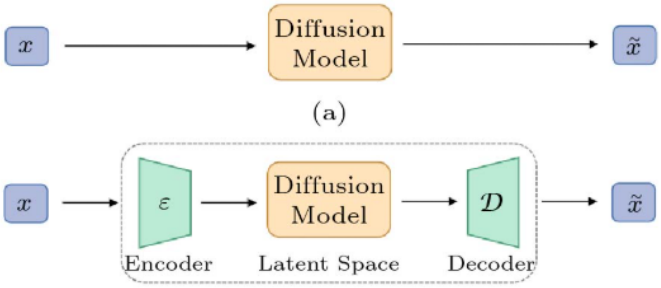

Figure 6: The architecture of an LDM compared to a standard DM.

for image generation. A particularly popular model of this category is Stable Diffusion which we will discuss bellow.

For our project, we leverage Stable Diffusion's core components:

- **CLIP text encoder:** Maps textual prompts (in our case figure captions) to embeddings that condition generation.
- **VAE:** Compresses images to/from latent space.
- **U-Net:** Predicts noise in latent space across timesteps.

All the loaded components are pretrained since training them from scratch would require tremendous compute. We first evaluate the pretrained model's zero-shot performance on scientific figure synthesis using domain-specific prompts, testing its ability to render schematics, graphs, and diagrams. To address observed shortcomings (e.g., label inaccuracies, flawed geometries), we then fine-tune the U-Net on a curated dataset of scientific figures, as presented in Section **??** , retaining the frozen VAE and text encoder for efficiency. This hybrid approach balances adaptation speed with the pretrained model's general knowledge.

3.2.2 Experiments with Pretrained U-Nets

To evaluate the impact of denoising timesteps, we test two pretrained U-Net configurations: one using a small sampling schedule (70 timesteps) and another

with the full schedule (1,000 timesteps). Increasing timesteps allows finer noise removal steps, improving output stability at the cost of slower generation.

- **U-Net with 1,000 timesteps**: This configuration should produce sharp and more consistent outputs by iteratively refining details across extended denoising steps. Indeed, when compared to the 70 timestep configuration in Figure 7, the generated images appear sharper, with grids in particular being much more consistent and smaller words (though hallucinated) appearing clear. Despite these improvements, the images are still unsuitable for scientific literature. Firstly, the model lacks consistency, generating various patterns that resemble figures and graphs but are ultimately illegible as a whole. Secondly, it fails to exhibit high-level reasoning, rendering it incapable of inferring the logical structure required for valid scientific images. For instance, in the learning curve example, where one would expect two distinct curves in a graph adhering to a specific form, the model does not even generate the background grid correctly, let alone the curves themselves. Lastly, the generated text is an incoherent mass that lacks structure and is scattered across the images, failing to convey meaningful information.

- **U-Net with 70 timesteps**: This model suffers from the same shortcomings as the previous configuration, as seen in Figure 7. The only notable difference is that it generates figures significantly faster but at the cost of increased artifacts. Due to the coarser noise removal, the outputs exhibit more noticeable distortions, such as blurry labels and misaligned geometries, further reducing their usability for scientific literature.

3.2.3 Performance Analysis

Let's analyze now the performance of the pretrained U-Net by plotting the values of the norm of latents at all timesteps during inference, as see in Figure 8. In this representative example we notice that the latent norms decrease almost linearly (without dropping massively however) from the initial random latent vector at timestep 0. This behavior is expected due to how the reverse diffusion process works. Initially, the latents start as pure Gaussian noise, which has a high norm. As denoising progresses, the model gradually removes noise while shaping the latents into structured features that resemble the target image. This leads to a reduction in overall variance and a decrease in norm magnitude.

As stated, this representative example closely resembles most other generated images. However the first column of images generated in Figure 7 (the bar plots), actually show a steady increase in the latent norm as timesteps progress as seen in Figure 9. This behaviour likely occurs because of improper **guidance scaling**.

In this work we utilized a technique called **classifier-free guidance** which determines how much the conditional prediction influences the final denoising step. This helps diffusion models steer the generation process toward more

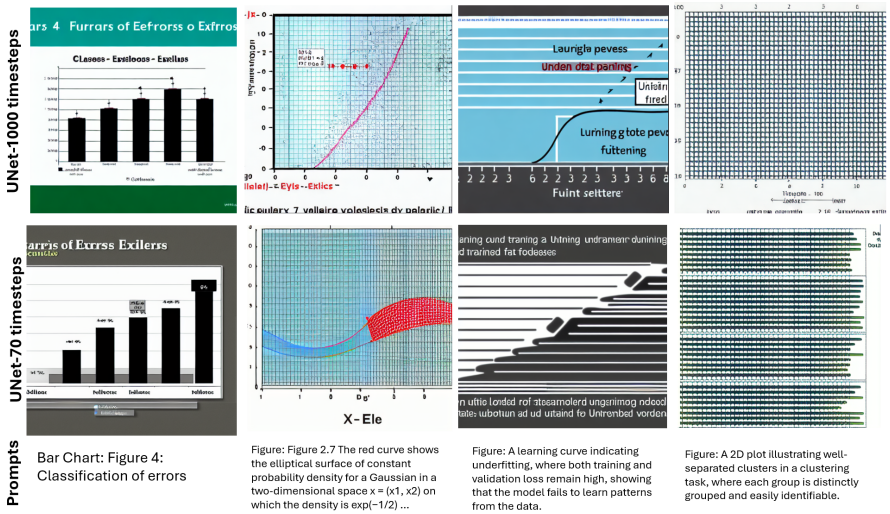

Figure 7: Generated images by a pre-trained Stable Diffusion model with different numbers of timesteps during inference.

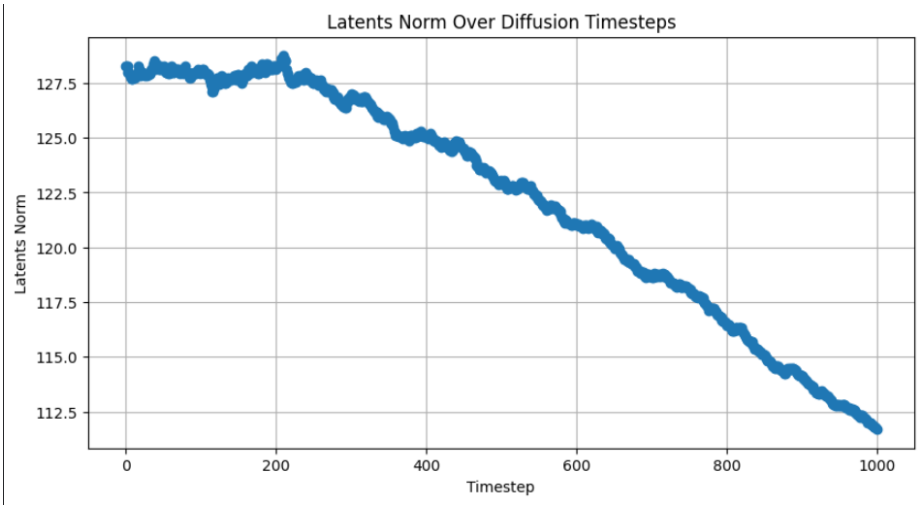

Figure 8: Representative example of latents norm over the span of inference.

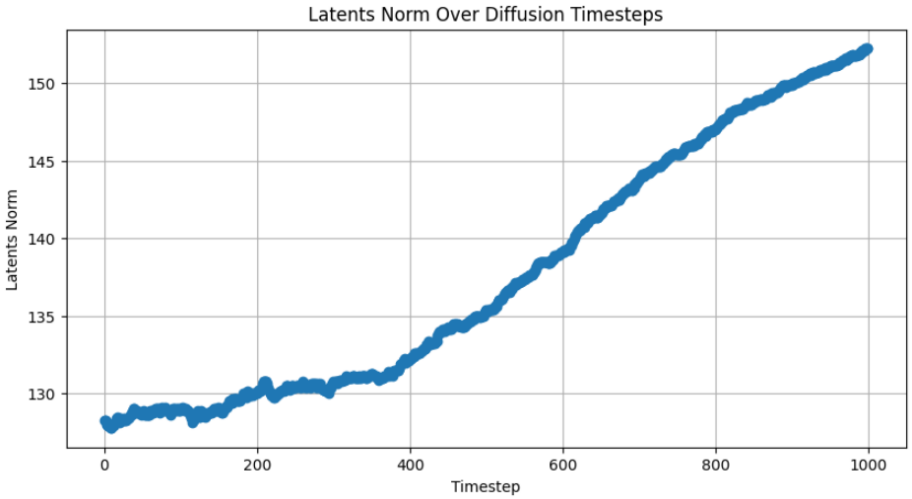

Figure 9: Outlier example where the latents norm increases steadily instead of decreasing.

meaningful outputs by adjusting the predicted noise residuals. The U-Net predicts two components, u and t, which represent unconditional and conditional noise estimates, respectively. The final prediction, is computed as

$$u + g \cdot (t - u)$$

, where g is the guidance scale. A higher g strengthens the influence of t, encouraging the model to follow the conditioning signal (e.g., figure caption) more strictly. This can improve coherence but may also lead to artifacts or mode collapse if set too high. Conversely, a lower g results in more diverse but potentially less relevant outputs. In our experiments a guidance factor of **7.5** seemed ideal for most cases. In some cases, like 9, it caused the latents norm to increase but the final generated image was still improved and more relevant to a scientific figure, than if guidance wasn't as strong.

3.2.4 Experiments with Finetuned U-Nets

For our first experiment we attempted to finetune the entire U-Net but quickly realised it was very computationally expensive (one epoch taking 5 hours of compute). To address these issues and accelerate training, we employ **Parameter-Efficient Fine-Tuning (PEFT)**, a family of methods that update only a small subset of a model's parameters during adaptation. Unlike full fine-tuning, which modifies all weights in a network, PEFT techniques such as layer freezing or adapter modules minimize memory usage while preserving the pretrained model's general capabilities.

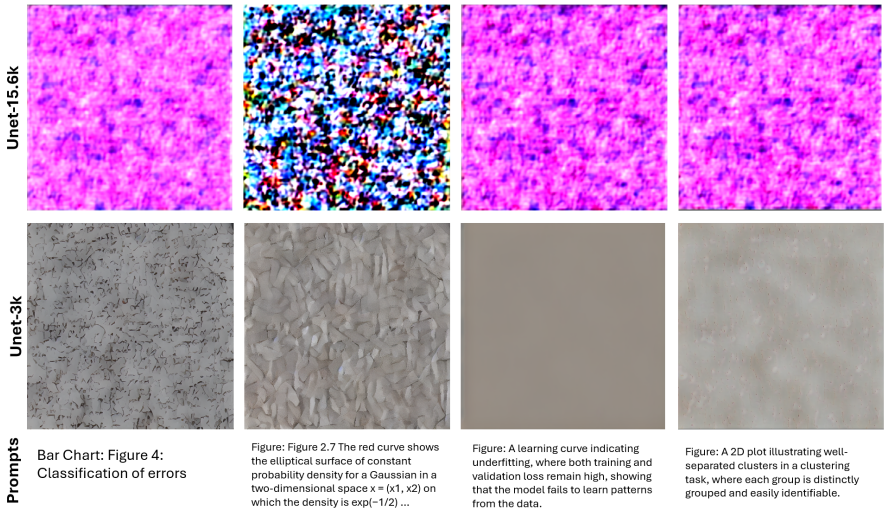

Figure 10: Generated images using finetuned UNet models.

We specifically adopt **Low-Rank Adaptation (LoRA)**, a PEFT method that approximates weight updates through low-rank matrix decomposition. During fine-tuning, LoRA freezes the original pretrained weights of the U-Net and injects trainable *low-rank matrices* alongside them. These matrices, enable adaptation to new tasks with a small number of parameters. Formally, for a pretrained weight matrix $W \in R^{d\times k}$, LoRA represents its update as:

$$\Delta W = BA \quad where \quad B \in R^{d\times r},\ A \in R^{r\times k},\ r \ll d, k \tag{1}$$

Here, r denotes the *rank* of the decomposition, controlling the trade-off between adaptability and efficiency. By training only B and A (containing a tiny fraction of the original parameters), LoRA achieves comparable performance to full fine-tuning while drastically reducing memory requirements.

Indeed, after utilizing LoRA, the number of total trainable parameters diminished (from 681m parameters to only 1.5m trainable parameters, approximately 0.2% of the total number). This decrease also led to a significant speed up in the training process (2x faster training speed, $\tilde{2}.3$ hours per epoch).

Having outlined the fine-tuning methodology employed in this work, we now present two experimental setups:

- **Fine-tuning U-Net on approximately 15.6k images for three epochs with 1000 timesteps in the scheduler:** This experiment involves extensive fine-tuning, utilizing the entire training dataset. The high number of timesteps is expected to enhance the generation fidelity of the diffusion model, but at the cost of increased computational complexity during inference. Training required approximately 2.3 hours per epoch, resulting in a total training time of approximately 6.5 hours for the complete

three-epoch process. However, despite theoretical expectations, the model exhibited signs of mode collapse after training. The generated images were predominantly noisy and blurry with minimal discernible structure, as can be seen in the first row of images in Figure 10. The only consistent modification observed was a shift in hue, which remained far from the intended output.

There are two primary reasons that likely contributed to this failure observed in our model. First, the dataset consists of highly domain-specific scientific figures that require a deep understanding of image semantics. Since the UNet was originally pretrained on diverse, natural images, generating complex scientific figures lies beyond its initial capabilities. Second, the dataset suffers from **severe caption ambiguity**. The captions often lack sufficient detail, resulting in **weak text-image alignment**. To mitigate this, we incorporated additional metadata, such as figure type, into the captions, but the information remained insufficient. For example, the caption **"Figure: Error rates of A"** provides no indication of whether A represents a model, a mathematical function, or something else, nor does it specify how the error rates are visualized (e.g., as a bar chart, line plot, or heatmap).

Given these challenges, we hypothesize that the model suffered from **early mode collapse**, where it overfitted to a subset of dominant patterns (e.g., specific colors and shapes) rather than capturing the full diversity of figures. This explains the **blurry, low-diversity outputs with a dominant hue** that we observed. To address this issue in future work, **longer training with a learning rate scheduler** could help stabilize optimization. Additionally, **dataset augmentation** would be essential. Generating more **informative captions** using an image-to-text model could provide a richer description of each figure, potentially improving **text-image alignment** and reducing ambiguity.

- **Fine-tuning U-Net on 3k images for three epochs with 70 timesteps in the scheduler:** As discussed earlier, mode collapse could be mitigated with extended fine-tuning and an improved dataset. However, due to computational constraints and the fact that creating a more robust dataset is beyond the scope of this project, we instead opted for a different approach. We trained the UNet again, this time using a smaller subset of images, aiming to prevent collapse while still encouraging the model to generate images with a more scientific appearance. Unfortunately, this approach also resulted in early mode collapse, as can be seen in the bottom row of images in Figure 10. Interestingly, this time the model exhibited **a bias towards a general gray hue**, with generated patterns appearing **smooth** and containing repetitive textures. Despite differences in the specific patterns learned, it is evident that in both cases, the model overfitted to certain colors and shapes while **failing to capture the higher-level semantics of the images**.

3.2.5 Performance Analysis

During inference, the two fine-tuned U-Net models exhibited distinct latent norm behaviors, reflecting different failure modes. The first model, trained on 15.6k images, steadily increased its latent norms over timesteps, as seen in Figure 11, suggesting an accumulation of dominant patterns rather than effective noise removal. This led to highly saturated, blurry, and noisy outputs, likely due to overfitting to certain color and texture distributions. In contrast, the second model, trained on 3k images, showed a steady decrease in latent norms, as seen in Figure 12, indicating a collapse towards low-energy states. This resulted in overly smooth, grey outputs with minimal detail, characteristic of mode collapse. Both cases demonstrate a failure to generalize, with one model amplifying certain features excessively and the other converging to an overly simplistic mean output.

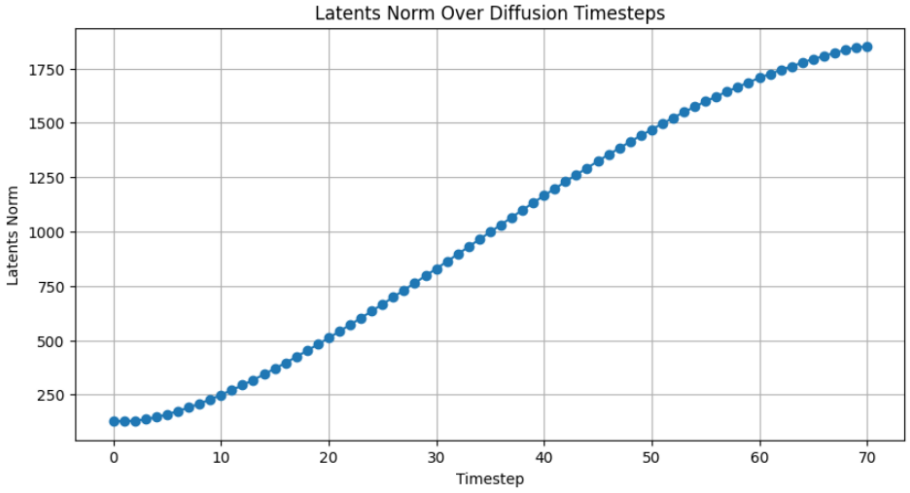

Figure 11: The latents norm is observed increasing steadily during the inference of the finetuned U-Net on 15.6k images.

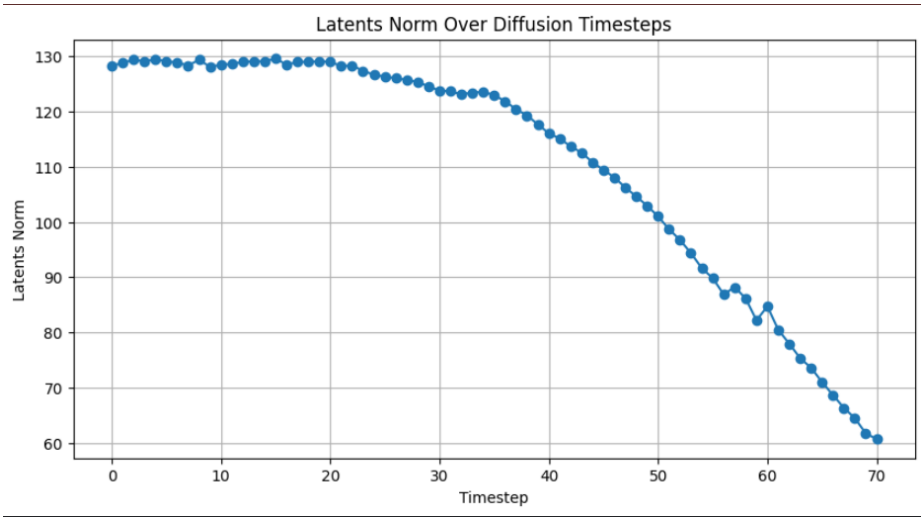

Figure 12: The latents norm is observed steadily decreasing during the inference of the finetuned U-Net on 3k images.

3.3 Vision Transformer-based Figure Synthesis

Vision Transformers (ViTs) are deep learning models that process images by dividing them into non-overlapping patches and treating each patch as an input token, similar to words in natural language processing transformers. Unlike convolutional neural networks (CNNs), which rely on local receptive fields, ViTs use self-attention mechanisms to capture both short- and long-range dependencies across an image, allowing for better spatial alignment and structural coherence. This makes them particularly well-suited for scientific figure synthesis, where precise geometric arrangements and text readability are essential. However, ViTs typically require large-scale training data to generalize effectively, and their performance can be limited on small datasets. Given the instability and structural inconsistencies observed in cGANs for figure generation, we explored ViT-based autoencoders to extract structured representations of figures and reconstruct them with improved coherence, aiming to achieve higher fidelity in scientific visualizations.

3.3.1 Experiments on Vision Transformer-Based Figure Generation

To assess the effectiveness of Vision Transformers in figure synthesis, we trained a ViT-based autoencoder on a dataset of extracted research figures. Our evaluation focused on three key objectives:

- **Structural Accuracy**: Determining whether generated figures retain geometric precision, text alignment, and visual coherence.

- **Feature Representation**: Evaluating the latent space organization and its impact on figure diversity.
- **Reconstruction Fidelity**: Measuring the ability of the model to faithfully reproduce input figures while maintaining clarity.

Model Architecture and Training Considerations For our experiments, we employed the `vit_small_patch16_224` model from the `timm` library as the backbone of our encoder. This model was selected due to its relatively small size, making it more suitable for limited computational resources while still capturing essential structural features.

The encoder processes images by dividing them into patches of size 16×16, which are then linearly projected into a sequence of embedded feature vectors that serve as input to the Transformer layers. This patch-based approach allows the model to capture global contextual relationships while maintaining computational efficiency.

Key Modifications and Considerations:

- **Removing the classification head**: Since the model is not used for classification, the output layer was replaced with an identity function.
- **Normalization in the latent space**: A Tanh activation function was applied to keep latent representations within a bounded range.
- **Efficient memory usage**: We enabled `cudnn.benchmark` and `cudnn.enabled` to optimize performance during training.
- **Image Preprocessing**: To ensure consistency and compatibility with the model's input requirements, all images were resized to 224×224 pixels. This standardization allowed for uniform processing across different figure types. Additionally, the images were converted into numerical tensor representations, which facilitated efficient manipulation and training. Finally, normalization was applied to scale pixel values within a fixed range, improving model stability and convergence during training. These preprocessing steps ensured that the model could effectively learn structural patterns from the dataset without being affected by variations in input dimensions or intensity distributions.

For the decoder, we implemented a transposed convolution-based architecture to upsample the latent representations back into image space. The main components of the decoder include:

- **Fully connected layer for reshaping**: The latent vector is mapped to a spatial representation ($512 \times 7 \times 7$) before upsampling.
- **Instance Normalization**: Instead of Batch Normalization, we used Instance Normalization to improve stability and adaptability to varying batch sizes.

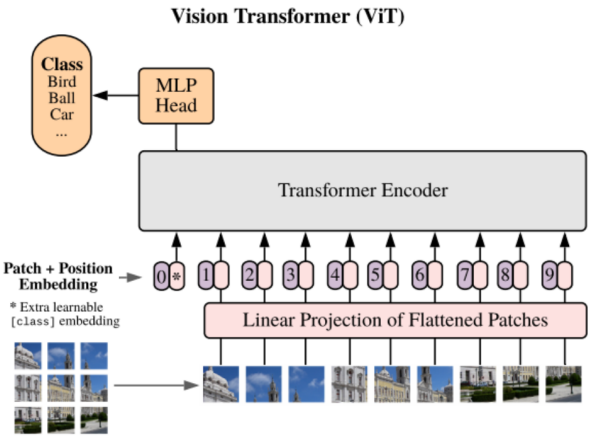

Figure 13: Visualization of Vision Transformer-based figure synthesis model.

- **Sigmoid activation in the final layer**: Unlike previous implementations that used `Tanh`, we opted for `Sigmoid` to constrain pixel values within the valid range of $[0, 1]$.

Despite these optimizations, our model exhibited clear limitations due to its relatively small scale. The encoder extracts only a compressed representation of the input figures, leading to some loss of fine-grained details in the reconstructed outputs. Additionally, due to the shallow depth of the decoder, certain complex structures were not fully recovered, resulting in blurred or abstracted features in the generated images.

Feature Mean and Standard Deviation Analysis The statistical distribution of extracted features from the autoencoder provides insight into the stability and effectiveness of the model. The computed values were:

- **Feature Mean**: 0.0264
- **Feature Std**: 2.4712

These values indicate that the model is capable of maintaining feature consistency but still exhibits significant variance, suggesting areas for potential improvement in regularization and representation control.

Encoder Feature Distribution To further analyze the encoded representations, we visualized the distribution of latent features extracted by the Vision Transformer encoder. The histogram of the latent space (Figure 14) reveals a Gaussian-like distribution centered around zero, with most feature activations falling within the range of -10 to 10. This suggests that the autoencoder effectively compresses figure representations into a structured latent space.

However, despite this structured encoding, the feature variance remains relatively high (Std: 2.4712), which may contribute to inconsistencies in the reconstructed images. The spread of feature values indicates that while the model learns meaningful embeddings, some latent variables capture unnecessary variations, leading to blurred or abstract reconstructions.

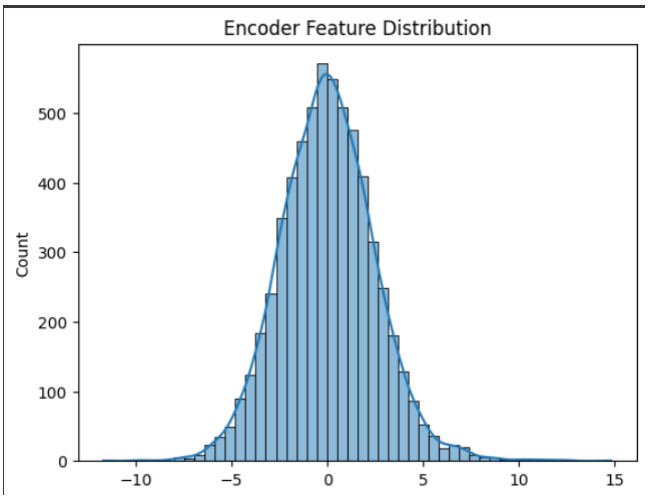

Figure 14: Histogram of the encoder feature distribution, showing a Gaussian-like spread centered around zero. The variance indicates the range of learned feature representations.

Visual Evaluation of Generated Figures After training the Vision Transformer model, the generated figures were visually inspected for structural consistency and category alignment. The results, while improved over cGAN-based approaches, revealed several persistent challenges:

- **Blurry and Diffuse Structures**: Many generated images retained vague approximations of figure components but lacked sharp edges and distinct textual elements.
- **High Contrast but Poor Detail**: Some figures featured strong black-and-white contrasts but failed to capture finer details such as axis labels and structured text.
- **Incomplete or Abstract Forms**: Several outputs exhibited abstract representations of scientific figures, where the general layout was present but precise elements were missing or distorted.
- **Feature Smudging and Artifacts**: Certain figures contained overlapping structures with unclear boundaries, suggesting that the model struggles to define individual components effectively.

Representative samples of ViT-generated figures are shown in Figure 15. These examples illustrate the strengths and limitations of Vision Transformers in structured figure synthesis.

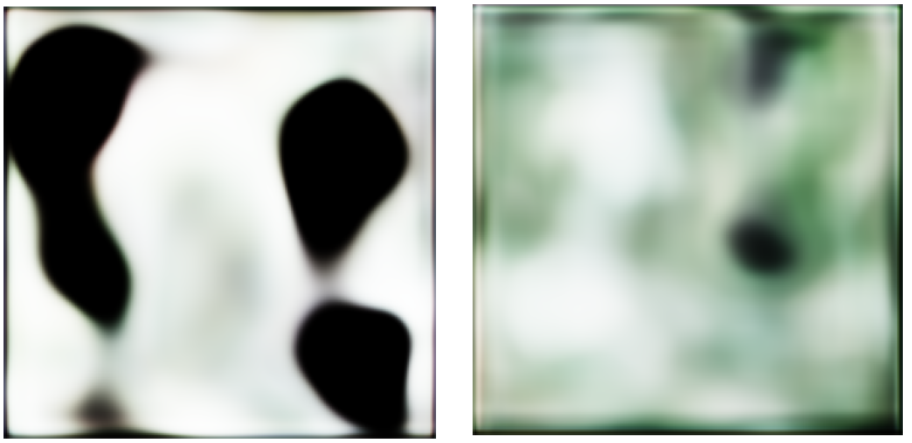

Figure 15: Comparison of generated figures using the Vision Transformer-based model. Left: `Bar chart`, Right: `Plot` , demonstrating structural coherence but revealing challenges in fine-detail reconstruction.

Comparison Between Original and Reconstructed Images To further assess the reconstruction capabilities of the Vision Transformer-based autoencoder, we compared original images from the dataset with their corresponding reconstructions.

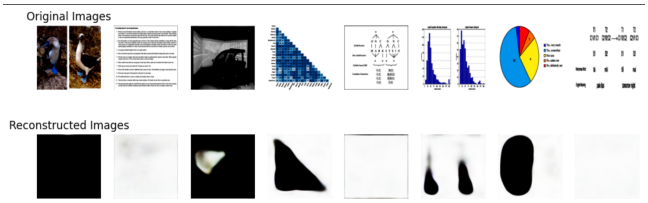

Figure 16: Comparison of original images (top row) and reconstructed images (bottom row) using the Vision Transformer autoencoder. The results indicate improved spatial alignment but reveal challenges in preserving fine details.

The model was evaluated using a subset of images from the dataset:

- The original images exhibited clear structures, including text and grid elements.
- The reconstructed images retained most of the essential structural details, though minor blurring was observed in high-frequency regions.

- Some reconstructions failed to capture sharpness, resulting in softened or smeared elements, suggesting a need for additional refinement in the decoder architecture.

These results further support the claim that Vision Transformers provide a robust framework for structured figure synthesis, offering substantial improvements over adversarial models.

Quantitative Performance Analysis To further assess the effectiveness of the ViT-based model, we computed the following quantitative metrics:

- **Inception Score (IS)**: Evaluates the diversity and realism of generated figures by analyzing classifier confidence.
- **Fréchet Inception Distance (FID)**: Measures the distributional similarity between real and generated figures.

Results and Observations The computed evaluation metrics highlight the strengths and remaining limitations of the ViT model in structured figure synthesis:

- **IS Score Improvement**: The ViT model achieved an Inception Score of $\approx 4.12 \pm 0.15$, significantly higher than the cGAN's 1.47 ± 0.10, indicating improved diversity and category differentiation.
- **Lower FID Score**: The ViT model attained a Fréchet Inception Distance of ≈ 135.3, a major reduction compared to the cGAN's 315.0, demonstrating much stronger alignment with real scientific figures.

Even after extended training (15 epochs), the ViT model demonstrated some improvements in structural alignment, but the generated figures still lacked clarity and fine-grained details. The reconstructions remained blurry, with noticeable artifacts and distortions, particularly in areas requiring precise geometric shapes or text rendering.

Limitations Due to Dataset and Model Size Despite the improvements achieved with Vision Transformers, the generated images still lack realism and fine structural accuracy. This can be primarily attributed to two key factors:

- **Small Dataset Size**: The ACL-FIG dataset is relatively small, limiting the diversity and variability of figure structures the model can learn. Without a larger dataset containing more scientific figures, the model struggles to generalize across different visualization types.
- **Limited Model Capacity**: The current Vision Transformer autoencoder is small in scale, optimized for efficiency rather than high-resolution synthesis. Larger models with more parameters and refined architectural components would likely improve reconstruction quality but require significantly more training data and computational resources.

Given these constraints, the current training setup is insufficient to produce highly realistic images. Future work should explore larger datasets and more advanced model architectures to improve structural fidelity and clarity.

4 Figure Synthesis based on Code-Generation Techniques

4.1 Introduction to code-generation with LLMs

Large Language Models (LLMs) have shown significant potential in code generation, enabling automated programming and the synthesis of complex scripts. These models, built on transformer architectures, are trained on vast text corpora to understand and generate human-like text. When fine-tuned on structured programming languages, LLMs can generate code snippets based on textual prompts, making them a promising tool for scientific figure generation.

However, deploying large-scale models is often infeasible due to computational constraints. While state-of-the-art (SoTA) models like GPT-4o and DeepSeek V1 deliver exceptional performance, they demand substantial resources. Given the limitations of this project, we focus on smaller, open-source LLMs optimized for efficiency, evaluating their ability to generate LaTeX code for figure synthesis. By comparing their performance to SoTA models, we aim to understand the trade-offs between computational efficiency and output quality.

A major constraint in our setup is GPU memory, as we are restricted to 16GB VRAM on the free NVIDIA P100 GPUs available through Kaggle. To accommodate this, we use compact models (7–12 billion parameters) and employ lower-precision or quantized versions to fit within these hardware limitations.

4.2 Experiments with various LLMs

To efficiently identify the most suitable models for our task, we conduct an experiment evaluating the pre-trained LaTeX code generation capabilities of several candidate LLMs. Each model is given a standardized, concise yet complete prompt and is tested on four different scientific figure captions. These captions, previously used in the image-generation experiments, vary in complexity, length, and clarity—ranging from straightforward and descriptive to highly ambiguous and intricate. This controlled evaluation provides a rough but insightful assessment of each model's familiarity with LaTeX syntax, reasoning ability, and capacity for generating structured and complex code. Based on these results we will proceed with the most promising model in the following sections to explore how we can further improve its LaTeX-generation ability through prompt engineering techniques and parameter-efficient fine-tuning.

We evaluate the models using two empirical metrics. The first is the number of valid LaTeX code snippets generated, with a maximum score of four, corresponding to the four test captions. A snippet is considered valid if it compiles without errors and does not produce an empty or obviously incorrect output.

Model	of Parameters	Precision	Valid Code	Correct Figure
Gemma 2 - it	9 billion	4-bit	0	0
Mistral Nemo Instruct	12 billion	4-bit	2	1
DeepSeek R1 Gwen	7 billion	4-bit	1	1
DeepSeek R1 Gwen	7 billion	16-bit	2	2
GPT-4o	˜1.73 trillion	32-bit (assumed)	4	4
DeepSeek V1	685 billion	32-bit (assumed)	4	4

Table 1: Empirical comparison of several LLMs on the task of LaTeX code generation.

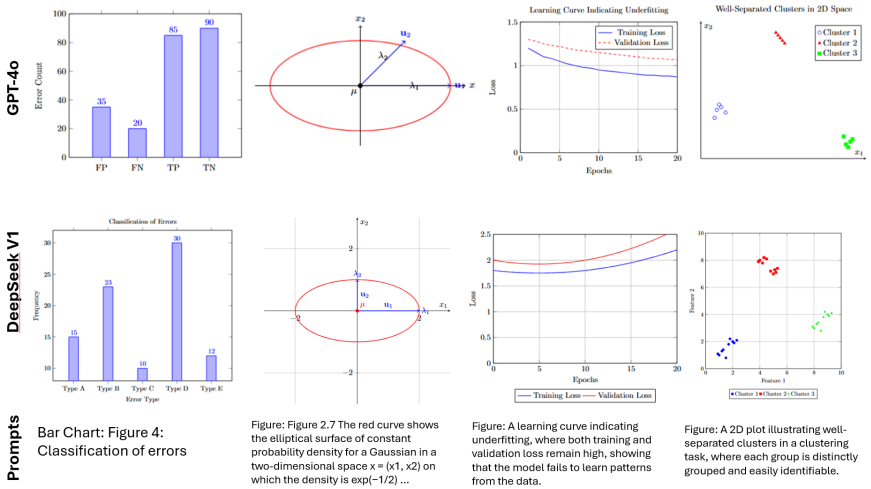

Figure 17: SoTA LLMs tested for Scientific Figure Synthesis

The second metric is the number of correct LaTeX figures, defined as those that both compile successfully and accurately represent the semantics of the given caption. Due to the inherent ambiguity of scientific figure captions, correctness is assessed based on human judgment rather than a predefined ground truth. For this reason, automating the evaluation of these models is not feasible. Even with a dataset of caption–LaTeX code pairs, no single figure is a unique solution to its caption. As a result, standard text similarity metrics like BLEU are ineffective, and image similarity techniques comparing the generated figures to a ground truth cannot be reliably applied.

In Table 1, we observe that the large models (GPT and DeepSeek V1) significantly outperform the smaller ones, achieving perfect scores by correctly generating figures for each given prompt. The generated figures are visualized in Tables 17 and 18 (excluding Gemma 2, which failed to produce compilable code, and DeepSeek R1 Gwen at 4-bit precision, as it performed better at 16-bit precision). These figures further highlight the varying capabilities of the

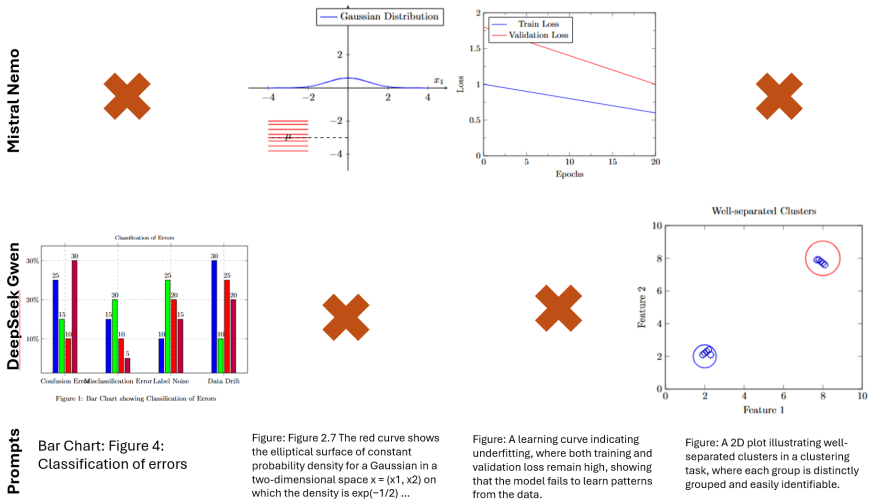

Figure 18: "Small" LLMs tested for Scientific Figure Synthesis

models. Among the smaller models, Gemma 2 underperforms compared to its competitors, while Mistral Nemo Instruct and DeepSeek R1 Gwen show promising results.

Initially, all models were quantized to 4-bit precision to accommodate the large sizes of Mistral Nemo and Gemma 2. However, for the relatively smaller 7-billion-parameter DeepSeek R1 Gwen, we were able to run inference at 16-bit precision. The results indicate that at 4-bit precision, Mistral Nemo performs best, but when tested at full 16-bit precision, DeepSeek Gwen outperforms it, suggesting that quantization-induced precision loss negatively impacted DeepSeek's generation capabilities. Given DeepSeek's strong performance and its comparably smaller size, we will proceed with it in the following experiments involving prompt engineering and fine-tuning.

4.3 Prompt Engineering for Improved Generation

Prompt engineering plays a crucial role in optimizing the performance of Large Language Models (LLMs), enabling them to generate more accurate and relevant outputs. By carefully crafting prompts, we can guide LLMs to better understand tasks, leverage in-context learning, and improve reasoning. In this study, we employ several prompting techniques to enhance LaTeX code generation for scientific figures: zero-shot prompting, guided few-shot prompting, example-based few-shot prompting, and chain-of-thought prompting.

The motivation behind these techniques is as follows:

- **Zero-shot:** This serves as a baseline for the model's performance without any additional context. Given that we work with small LLMs, a minimal-

Model	Prompt	Valid Code	Correct Figure
DeepSeek R1 Gwen	Zero-shot	2	2
DeepSeek R1 Gwen	Guided few-shot	2	1
DeepSeek R1 Gwen	Example-based few-shot	3	3
DeepSeek R1 Gwen	Chain-of-Thought	3	1

Table 2: Empirical comparison of different prompting techniques on DeepSeek Gwen.

token prompt may be easier for the model to process and less likely to interfere with its inherent reasoning patterns.

- **Guided few-shot:** This prompt includes a few caption/LaTeX code pairs presented in a text-completion format, guiding the model toward generating structured LaTeX code. Through in-context learning, the model may better infer the desired output style and structure. This technique is particularly useful when the pre-trained knowledge of the model lacks specific details necessary for LaTeX-based figure synthesis.

- **Example-based few-shot:** Similar to guided few-shot prompting, this approach provides multiple caption/LaTeX code pairs but frames them as examples rather than enforcing a completion-style format. Since DeepSeek generates responses step-by-step by default, allowing it to process examples without forcing a strict text-completion pattern may preserve its learned reasoning chain and lead to more natural LaTeX generation.

- **Chain-of-Thought:** This method is designed to enhance reasoning by explicitly prompting the model to generate intermediate logical steps before producing the final output. Interestingly, DeepSeek already performs a stepwise reasoning process without explicit instruction. Thus, this experiment primarily aims to investigate whether explicitly prompting for a chain-of-thought will enhance or disrupt its existing reasoning pattern, potentially affecting the quality of generated figures.

These different prompting strategies allow us to systematically analyze how various forms of guidance impact the model's ability to generate accurate LaTeX representations of scientific figures.

4.4 Finetuning for Improved Generation

4.4.1 Custom Dataset of LaTeX figures and their captions.

To begin the fine-tuning process, we require a dataset specifically tailored for training the DeepSeek Gwen model. Given that GPT-4o demonstrated strong performance in LaTeX-based figure generation, we leveraged it to construct a synthetic dataset consisting of LaTeX code and corresponding descriptive captions. The model was explicitly instructed to generate figure/caption pairs

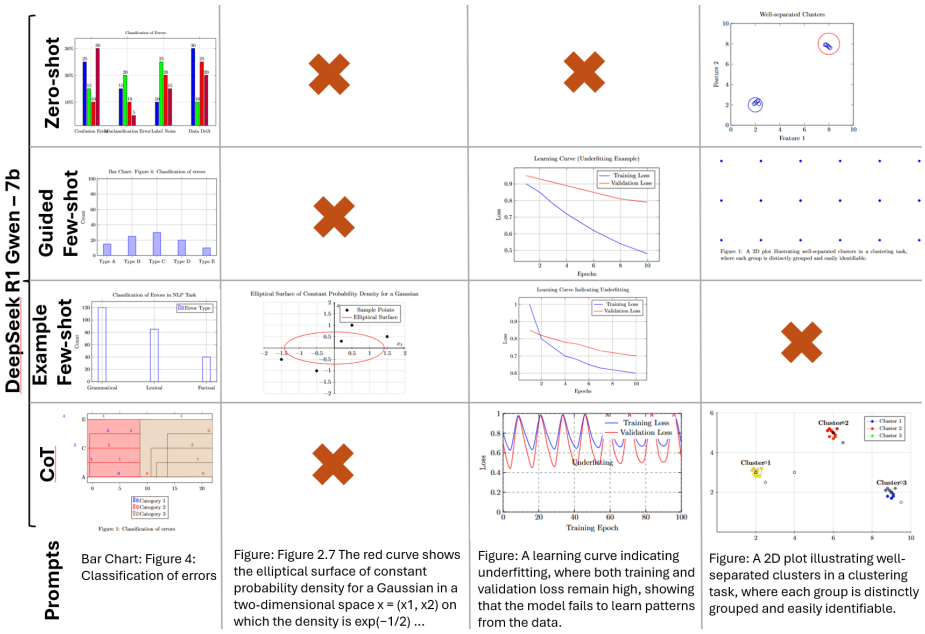

Figure 19: Enter Caption

centered around Machine Learning, ensuring a diverse range of figures in terms of complexity, from simple to intricate designs, as well as variation in caption clarity, ranging from highly descriptive to more ambiguous statements.

Each generated figure/caption pair underwent manual evaluation before its inclusion in the dataset to ensure quality and relevance. Due to time constraints, the final dataset comprises only 40 figure/caption pairs. While LLMs are generally capable of adapting their behavior with very limited training samples, a dataset of this size may still be insufficient for effective fine-tuning. However, it provides an initial foundation for exploring how the model will behave after being finetuned to this specific task.

4.4.2 Finetuning Method

Similar to our previous experiments with Stable Diffusion, fine-tuning this LLM requires Parameter-Efficient Fine-Tuning (PEFT) methods due to its large size of 7 billion parameters and our limited hardware (16 GB of VRAM). To achieve this, we employ the same LoRA (Low-Rank Adaptation) technique as before, which enables efficient fine-tuning by modifying only a small subset of model parameters while keeping the majority frozen. Additionally, we apply quantization, reducing the model's parameter precision down to 4 bits, significantly decreasing memory requirements. By combining LoRA with quantization (and other techniques like gradient checkpointing), we can effectively fine-tune a 7-billion-parameter LLM on a single 16 GB VRAM GPU, making the process

feasible despite hardware limitations.

For the fine-tuning process, we provide the model with a structured input consisting of a simple prompt describing the task, followed by a caption from the dataset and its corresponding LaTeX code. This setup helps the model better understand its objective while also familiarizing it with LaTeX syntax for figure generation.

While all training samples are correct, it is important to note that the model's loss is computed based on string dissimilarity between the generated LaTeX code and the reference code from the dataset. However, a given caption does not have a unique figure solution, and even a specific figure can be represented using multiple valid variations of LaTeX code. Consequently, the current loss function may not be ideal for optimizing figure generation. We note that a more sophisticated loss function, capable of accounting for semantic correctness rather than mere string similarity, would likely improve training outcomes, unfortunately such a function is beyond the scope of this project.

4.4.3 Experiment Results

After fine-tuning our model, we evaluate it using the same set of four captions as before. Additionally, we assess its performance across different prompting strategies, including the four prompts tested in the **Prompt Engineering** section, along with an additional prompt referred to as the **finetuned prompt**. This new prompt mirrors the structure of the input used during training and serves as a simplified version of the zero-shot prompt, designed to minimize token count for efficiency and clarity during fine-tuning.

In theory, if the model has effectively adapted to the few training examples provided, it should perform best with the finetuned prompt, as it has learned to associate it more directly with LaTeX figure generation. However, this prompt structure might suppress the model's natural step-by-step reasoning process, which it typically employs by default. Thus, this experiment not only evaluates whether slight fine-tuning enhances performance but also investigates whether it disrupts the model's reasoning, leading to a decline in generation quality. Additionally, we examine whether fine-tuning affects the model's performance with the original four prompts, examining whether the learned patterns generalize or overfit to the training distribution.

We present the evaluated experiments with the same metrics as before, the total count of valid codes generated and the total count of correct figures. The results can be seen in Table 3.

The results suggest that the model did not effectively adapt to the training examples, likely due to the limited dataset size. Consequently, the finetuned prompt (expected to outperform other prompting strategies) failed to generate a single correct figure in our four test cases. This outcome also supports our hypothesis that slight fine-tuning on a model that inherently performs step-by-step reasoning may disrupt its thought process. Rather than improving generation quality, fine-tuning appears to have hindered the model's reasoning, ultimately leading to worse performance.

Prompt	Valid Code	Correct Figure
Zero-Shot	2	0
Guided Few-Shot	3	1
Example-based Few-Shot	4	1
Chain-of-Thought	0	0
Finetuned Prompt	1	0

Table 3: Empirical comparison of different prompting techniques on our finetuned DeepSeek model.

Comparing the performance of the finetuned model across the different prompts to that of the base model, we observe that finetuning overall diminished the model's performance. The best-performing prompt remained the **example-based few-shot**, likely for the same reasons outlined in our prior analysis. However, prompts that previously yielded decent results, such as **zero-shot** and **chain-of-thought**, were negatively impacted by finetuning, leading to no correct figures being generated. This further reinforces the idea that slight finetuning on a step-by-step reasoning model may interfere with its original inference patterns rather than enhancing them.

After examining the model's generated outputs, this suspicion is further supported, as the finetuned model exhibits an inconsistent generation pattern. It often starts by generating LaTeX code, then abruptly shifts to a thought-process explanation before switching back to code, without maintaining a clear structure. This behavior was not observed in the base model prior to finetuning, further suggesting that the fine-tuning process disrupted its learned step-by-step reasoning rather than improving its performance.

4.4.4 Conclusion

After a thorough investigation into the potential of LLMs for Scientific Figure Synthesis, we have demonstrated their capability to generate highly relevant LaTeX code, effectively capturing even complex concepts. Our evaluation of multiple popular LLMs provided insights into their current strengths and limitations in LaTeX code generation. Based on these findings, we selected the DeepSeek R1 Gwen 7B model as the most promising candidate for further experimentation. We then explored various prompt engineering techniques to analyze how different prompting strategies influence the model's performance. Additionally, we generated a synthetic dataset and fine-tuned the model to assess whether targeted training could enhance its figure synthesis abilities. Notably, using an LLM for scientific figure synthesis not only automates the initial code generation but also offers flexibility (users can refine the generated LaTeX code to better suit their specific needs). Furthermore, we note that a future direction for this task could be an interactive, chat-based workflow. If explored, it could allow users to iteratively refine figures through dynamic feedback, potentially improving both usability and accuracy.

5 Conclusion

This project explored multiple generative AI approaches for scientific figure synthesis, evaluating their effectiveness in generating structured, high-quality figures. We examined three primary image-generation techniques: Generative Adversarial Networks (GANs), Latent Diffusion Models (LDMs), and Vision Transformer-based autoencoders. Additionally, we investigated Large Language Models (LLMs) for generating LaTeX code-based figures as an alternative after seeing the numerous disadvantages of the image-generating methods.

Our experiments demonstrated that while image-generation models can capture some structural patterns, specifically with pre-trained Stable Diffusion models, they consistently struggled with fine-grained details, text readability, and geometric precision. The primary limitations observed across these models included:

- **GAN-based Models**: Conditional GANs (cGANs) failed to generate meaningful scientific figures due to mode collapse and adversarial instability. The outputs were often noisy and lacked any proper semantics or structure. Despite extended training, cGANs were unable to generate images that resembled scientific figures.
- **Latent Diffusion Models (LDMs)**: Although diffusion models demonstrated stronger image synthesis capabilities than GANs, their inability to generate structured figures was still evident. Pretrained models such as Stable Diffusion could not create accurate scientific figures, often producing hallucinated elements and distorted text, while following the prompted caption very loosely. Fine-tuning attempts were computationally expensive and resulted in a mode colapse of the model. The small dataset size and lack of structured training data hindered their performance.
- **Vision Transformer-based Autoencoders**: The Vision Transformer model performed poorly in capturing high-level structural features, like both other methods. Its reconstructions remained blurry and lacked textual clarity. Despite improvements in spatial alignment and feature representation, the model's limited capacity and dataset constraints prevented it from achieving high-quality results. While the quantitative evaluation showed an improvement over adversarial models, the overall quality was still insufficient for practical scientific use.

In contrast, **code-generation techniques** using Large Language Models (LLMs) proved to be significantly more effective for scientific figure synthesis. By leveraging structured LaTeX generation, LLMs produced figures with exact precision, eliminating the issues of noise, distortions, and illegible text that plagued image-generation approaches. Furthermore, because of LLMs inherent reasoning capabilities they were able to generate figures that accurately modeled their sometimes ambigious captions. Our experiments with multiple LLMs highlighted:

- **Prompt Engineering**: Structured prompts greatly enhanced LaTeX code generation, improving figure accuracy and coherence.
- **Fine-tuning**: While fine-tuning on a small dataset of LaTeX code and figure captions did not yield significant improvements, pre-trained large-scale models (such as GPT-4o and DeepSeek V1) generated high-quality figures even without this extensive adaptation.
- **Flexibility and Customization**: Unlike image-generation models, code-generated figures could be manually refined and adapted, making them a more practical solution for researchers requiring high-precision visualizations or those looking for a basic LaTeX figure template that they can further refine on their own.

Ultimately, our study demonstrates that while deep generative models such as GANs, LDMs, and ViTs offer promise in structured image synthesis, they are currently ill-suited for scientific figure generation. The inherent challenges of text rendering, spatial consistency, and geometric precision make AI-driven image generation unreliable for this task. In contrast, code-based figure generation through LLMs presents a more practical and scalable approach, ensuring high accuracy and usability in academic and scientific communication.

Future Work: Future research should focus on integrating structured text-based figure generation with interactive user refinement, allowing AI-assisted workflows to streamline scientific visualization while maintaining accuracy and flexibility. One possible direction is improving LaTeX/caption dataset quality and size, as training on larger and more diverse scientific figures could help LLMs become better acquainted with LaTeX code and generate more accurate figures. For increased consistency in generating LaTeX code without the need for extensive fine-tuning, template-based prompts could be explored, where an appropriate template for a figure would automatically be selected, given an input prompt, and the LLM would have to 'fill in the gaps' and complete the template. Developing better similarity metrics for scientific figures is another crucial step, as current assessment methods focus primarily on string similarity, when finetuning LLMs, rather than structural and semantic similarity between generated figure and input caption.

To improve the accuracy of GANs, diffusion models, and Vision Transformers for scientific figure synthesis, several enhancements could be explored. GANs could benefit from architecture refinements and improved loss functions to mitigate mode collapse and enhance structural coherence. Diffusion models could be improved by fine-tuning on larger, more diverse datasets with explicit constraints on scientific figures, ensuring better adherence to structured layouts. For Vision Transformer-based autoencoders, increasing model capacity and introducing more sophisticated upsampling techniques could lead to enhanced results

Ultimately, more work needs to be done for image-generation based scientific figure synthesis, particularly focused on creating large datasests, investing in

extensive training for complex models and even investigating hybrid approaches that incorporate high-level textual semantics from LLM models directly into image-generating models to allow for better undestanding of the nuances in the input text caption. Currently, LaTeX code generation from LLMs seems to be the most promising approach for SciFS.

References

[DAGY+25] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan

Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

[Dos20] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[HSW+21] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[Mir14] Mehdi Mirza. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[RBL+22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[TRP+24] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder

Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024.

[XXQ+23] Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment, 2023.

[RBL+22] [DAGY+25] [TRP+24] [HSW+21] [XXQ+23] [Dos20] [Mir14]