Guided Conditional Image Generation with Conditional Flow Matching

Anshumaan Chauhan
achauhan@umass.edu

Eric Anderson
edanderson@umass.edu

Ashvath Balgovind
abalgovind@umass.edu

Abstract

In this paper, we are focusing on the task of Conditional Image Generation using the Flow Matching (FM) with Optimal Transport (OT) objective. We propose an attention incorporated UNet model, which utilizes Classifier-Free Guidance (CFG) to learn the task of conditional and unconditional image generation simultaneously. On evaluation, Unconditional FM model has an FID score of 105.54, whereas the Conditional FM model showcases FID score and CLIPScore of 385.86 and 22.19 respectively. (Github Link)

1. Introduction

Over the past few years, various deep generative models such as Generative Adversarial Networks (GANs) [3, 8, 9], and Variational Auto-Encoders (VAEs) [2] have shown the ability to generate high quality photo realistic images. During training, GANs and VAEs are not learning the exact marginal joint probability distribution over the real data (VAEs use the variational lower bound technique for approximation of the marginal joint distribution, and GANs aim to min-max the classification error). This results in low quality image generation by VAEs, and training challenges in GANs such as mode collapse and vanishing gradients. [19].

An image may be worth a thousand words, but it is still non-trivial to condition image generation with text. [20]. Previous work has used GANs and VAEs for text conditioned image generation[4, 11]. However, even with architectural modifications such as including attention mechanisms and adding new conditional information beyond text, they suffer with object distortion, incorrect positioning, and unnatural overlap scenarios [13].

Flow-based Models [14] were proposed in order to resolve the drawbacks suffered during the training of GANs and VAEs (related to not learning the exact joint probability distribution of the provided data). They use a series of functions/transformations (called flows) which transform a simple distribution to a complex distribution given that they satisfy the following conditions - i) are invertible and ii) have an easy-to-compute log determinant of the Jacobian matrix, in order to learn the exact probability distribution. Apart from the limited expressiveness due to these conditions, the calculation of the determinant of the Jacobian matrix becomes a bottleneck due to its cubic computational complexity.

Utilizing the instantaneous change of variables theorem, we are able to use the trace (which is of a linear nature) instead of the determinant of the Jacobian Matrix of Vector Field. Neural Ordinary Differential Equations (ODEs), when solved with this operation using ODE Solvers, lead to models called Continuous Normalizing Flows (CNFs) [1].

Unfortunately, calculating the trace is still a difficult operation when working in high dimensions. Therefore, Flow Matching (FM) [10] was proposed as an efficient simulation-free approach to train CNF models. However, the FM objective function contains some intractable integral terms. Surprisingly, it was showcased that optimizing the conditional flow matching (CFM) objective was equivalent to optimizing the FM objective. Flow Matching focuses on learning the vector fields that will constitute the flow of transition from one probability density function to another probability density function.

Employing FM with diffusion paths results in a more robust and stable alternative to training Diffusion models [7]. However, these diffusion probability paths tend to overshoot the final sample, and therefore we have to use backtracking as an additional step. Conditional Optimal Transport (OT) probability paths are shown to be superior to Diffusion Paths, as they provide faster training and sampling processes [10].

Our project implements a modified UNet [16] architecture trained with the CFM objective function to learn the Optimal Transport Vector Fields for Conditional Image Generation.

We analyze the effect of conditional parameters (textual embeddings that correspond to each image's label) on the training of a Flow Matching Model. Conditioning of the image based on the text will be done using Classifier Free Guidance, which will help develop a model that supports both conditional and unconditional image generation. More details about how we plan to include these text embeddings

into the learning process is discussed in Section 3.

Lastly, we implement a modified UNet model by incorporating different layers such as attention mechanisms, and experiment with hyperparameters such as the learning rate and depth of the model. To evaluate the model, we use CLIPScore, Fréchet inception distance (FID) and qualitative human evaluation.

2. Data and Evaluation Metrics

2.1. Data

We use the CIFAR-10 dataset which is publicly available at University of Toronto's website. The CIFAR-10 dataset consists of 60,000 colored images of 10 different classes. Each class consists of approximately 5000 images for training and 1000 images for testing. Images are of size 32x32x3 and they each fall under a one-word class label.

2.2. Performance Evaluation

CLIPScore [5] and Fréchet Inception Distance (FID) [6] are used as performance metrics to evaluate the model. CLIPScore measures the similarity between an image and a corresponding caption (utilizes the CLIP model for this task - values ranges between 0 and 100). A higher CLIPS score (close to 100) indicates high compatibility of the image with the caption. Fréchet Inception Distance measures the feature distribution between the generated images and the real images. Low FID score (close to 0) indicates that the high-level features of our model-generated images are more similar to high-level features in real images.

Both these scores are known to be highly correlated with human judgment.

3. Methodology

Our project consists of five main components, Data Augmentation, Timestep Encoding, Text Encoding, Training, and Inference.

3.1. Data Augmentation

The CIFAR-10 dataset as described in Section 2.1, consists of images and labels. Each label belongs to any one of the following 10 classes, {airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck}. These labels are suitable for an initial implementation of our project, however they miss a lot of information about the image. For example, given an image from the truck class, the label does not have any information of the background, relative size of the truck, etc.

To overcome this issue, we make use of an image-to-text transformer model BLIP2 Flan T5 XL. This visual transformer model helps us generate a Natural Language phrase for each CIFAR image, which gives a more detailed description of each image.

3.2. Timestep Encoding

Flow generation aims to calculate flow from a uniformly sampled image x_0 to a final output image x_1.

When training our model, we use linear interpolation to generate an input image that represents the image at timestep $t \in [0, 1]$. To incorporate this timestep information in the processing of the image, we utilize Sinusoidal Positional Encodings [18].

$$\text{Sinusoidal Encoding}(t, i) = \begin{cases} \sin(\frac{t}{10000^{2i/d}}) & \text{if } i \text{ is even} \\ \cos(\frac{t}{10000^{2i/d}}) & \text{if } i \text{ is odd} \end{cases}$$

where i is the dimension and t is the timestep.

After getting these Sinusoidal Time Encodings, we pass them through a Linear layer and a ReLU activation prior to passing this time representation into our down and up sampling blocks. Within each block, the time representation is passed through another GELU activation and a Linear layer, before being added into our image representation.

3.3. Text Encoding

In conditional image generation, we need to perform image conditioning using natural language text inputs.

To transform our natural language inputs into a form that the model can use, we use a BLIP2 Flan T5 XL tokenizer. We use these tokenized inputs to pass our language information through the model, which guides the model to condition its image generation on text inputs. To encode and incorporate the tokenized inputs, Cross-Attention has shown to be successful in many Language Vision Models [15].

Tokenized values are initially projected into a space of $C * H * W$ dimensions using a Linear layer, where C, H and W correspond to number of channels, height and width of the images respectively. To get the encoding values into a range similar to image pixels we perform a normalization operation. These projected normalized values are then reshaped into a matrix of size [N,C,H,W], where N is batch size. Convolutional operations are performed on the reshaped matrix to generate the key (K) and value matrices (V). Similarly, a convolutional operation is applied on the processed images to get a query matrix (Q). A scaled dot product is performed between K and Q to get the normalized attention scores, which are used along with the value matrix V to obtain our final text encoding. 1.

$$\text{Attention (Q,K,V)} = \text{Softmax}(\frac{Q * K^T}{\sqrt{d/h}}) * V \qquad (1)$$

3.4. Training of the Flow Matching Model

This section is the crux of our project, as it describes the process of training a Flow Matching model. Flow Matching

is the task of learning vector fields that help transition from one probability distribution to another. The entire problem can be simply framed as: we start from a Uniform probability and want to learn vector fields that guide us in the direction of true probability distribution of the data.

We start with an image that consists of uniformly sampled RGB values that has the same dimensions as our CIFAR images. The aim is to learn the flow of each pixel as it transforms from timestep $t = 0$ (Uniform Noise Image) to timestep $t = 1$ (the CIFAR iamge).

$$X_0 \sim \mathcal{N}(\mu, \sigma^2);\ .\quad \mu = 0, \sigma^2 = 1$$

$$X_1 \sim \mathcal{D};\ \ \mathcal{D} \text{ denotes the set of CIFAR images}$$

A new Uniform Noise image is generated at $t = 0$, which our FM model uses to learn the flow to a CIFAR image at $t = 1$; where y is the conditioned text (Refer to Equation 2).

$$x_t = x_0 + \int_0^t v(t:y)dt \quad \text{v(t) is the vector field} \qquad (2)$$

The Flow Matching objective is as follows:

$$\mathcal{L}_{CFM}(\theta) = \mathbb{E}_{t,q(x_1),p_t(x|x_1)}||u_t(x) - v_t(x|x_1)||^2 \qquad (3)$$

In equation 3, $t \sim \mathcal{U}[0, 1]$ (uniformly sampled timestep between 0 and 1), u_t represents the actual flow, where as $v_t(x|x_1)$ represents the conditional flow. x_1 is sampled from the true probability distribution $q(x_1)$, and $p_t(x|x_1)$ is the conditional probability path used to sample x (it is generated using $u_t(x|x_1)$). p_t approximates the probability distribution q at time t=1.

The actual flow from some point in space at time step t = t, is the average of all the flows between x_0 and x_1 that passes through that point. The optimal transport equation helps us define this actual transport vector field as $x_1 - x_0$. Intuitively, the optimal path from one location to another is a straight line, which is also the case for the Optimal Transport vector field. Therefore the OT vector field objective function is:

$$\mathcal{L}_{CFM}(\theta) = \mathbb{E}_{t,q(x_1),p(x_0)}||v_t(\Psi(x_0:y)) - (x_0 - x_1)||^2 \qquad (4)$$

where x_0 is the Uniform Noise images, t is the timestep, x_1 is the CIFAR images, $\Psi(x_0)$ is the conditional flow parameterizing the learned vector field v_t (conditioned on text y).

In Figure 1, we show the entire process of training a Flow Matching model. We start by generating Uniform Noise images for each corresponding image in the current batch of CIFAR images. However we also sample a timestep t for each pair uniformly between 0 and 1, and then linearly interpolate our Uniformly sampled random image using the formula $x_t = t * x_1 + (1 - t) * x_0$. The result of this linear interpolation is an image from the probability space of timestep t. This process helps in learning the flows at all timesteps.

Now we utilize equation 4 to get the loss value between the learned vector field and the actual vector field. Optimization of the weights is performed using backpropagation algorithm using Adam optimizer.

This entire training process is repeated for several epochs in batches, and finally we have optimal θ, parameters for our UNet model, which has successfully learned the vector fields between a random uniform distribution and the CIFAR dataset distribution.

3.5. Inference

In this section, we explain the process of generating an image resembling the distribution of CIFAR dataset, which is constructed from the text embeddings and random uniformly sampled noise.

We have seen that our model outputs the vector field v_t when it is provided with the Uniformly sampled image and the tokenized text embedding. But how do we get to a CIFAR image using this vector field? - we use ODE Solvers for this task. ODE Solvers take in the vector field function, time step, tokenized text and the image as input and give us the resulting image that follows the vector field to time step t=1.

The entire process of Inference using Flow Matching models is presented in Figure 2. Mathematics behind the working of ODE solvers is beyond the scope of this paper.

4. Implementation and Results

4.1. Implementation Details

A high level architecture design of the proposed UNet model is presented in Figure 3 (more details about the individual components are mentioned in the Appendix section - Figure 10, 11). A similar architecture was used for both conditional and unconditional generation, though the Cross-Attention block is not used during the training of unconditional generation. The proposed model has 191M parameters (∼ 729 MB), and was trained on the entire training set for 96 epochs with a batch size of 128. Learning rate was $1e^{-4}$ and $3e^{-4}$ for the unconditional and conditional generation respectively. For faster and offline training of the model, we utilized NVIDIA A100 GPU (on Google Colaboratory).

4.2. Image Captioning and Text Embedding

To condition our models on input text, we need a training set containing text phrases that describe each image. The CIFAR dataset does not contain text beyond one-word labels.

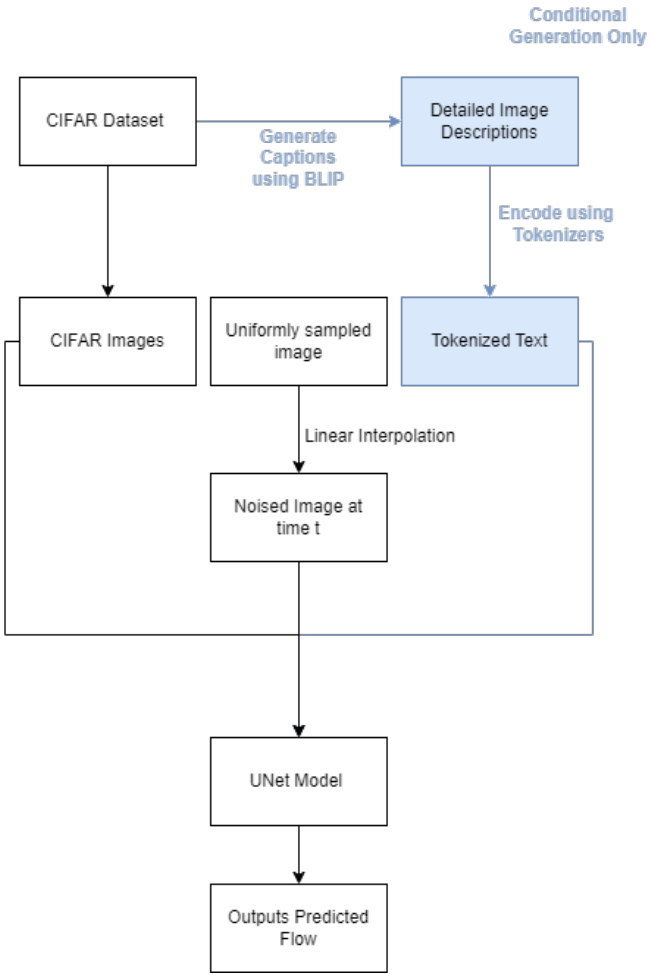


Figure 1. Training of the Flow Matching Model

We explored a couple of options for generating text phrases by running examples from CIFAR-10 through models capable of image captioning.

To measure the performance of each model, we randomly sampled 100 images and evaluated the accuracy of each model. For a generated caption to be considered accurate, we say that the caption must have information about the true label (or a more specific form of the label, for example, if the label is "dog" and the model mentions "German Shepherd" then it is acceptable) and the caption must not describe anything in the image that is not spatially present.

We evaluated the performance of VIT GPT2 [12] (Figure 9) and BLIP-FlanT5-XL [17] (Figure 4) models with a small random subset of 100 CIFAR images on the task of image captioning. An accuracy of 95% and 77% was observed for BLIP-FlanT5-XL and VIT GPT2 respectively. BLIP-FlanT5-XL, due to its capability of generating highly accurate captions, is being utilized to for the task of image captioning on entire CIFAR dataset.

One of the possible future works could be utilizing the CLIP model to generate match scores for the generated caption and the corresponding image. Image-caption pairs with match scores below a threshold value will be dropped from the dataset/ or will have their captions re-generated.

4.3. Experiments

4.3.1 UNet Model Variations

Throughout the experimentation phase, we made three major architectural changes to the base UNet model, which were based on the performance analysis of the prior model.

We started with a base UNet model, which was updated to incorporate sinusoidal time step encodings. This unconditional FM model was trained on a small (pseudo) dataset of 30 images for 2000 epochs with a batch size set to 10 and a learning rate of $3e^{-4}$. The image generations were not consistent with the training set, and were mostly composed of noise. Observed inconsistency was due to

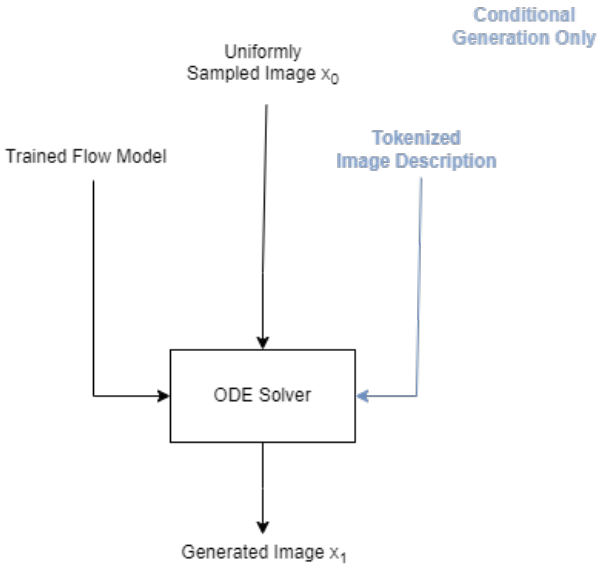


Figure 2. Inference using ODE Solver

the use of Batch Normalization after every convolutional filter, that caused normalization across images of different classes within each batch. To overcome this problem, we swapped Batch Normalization with Group Normalization. The change leads to normalization taking place over each independent channel in an image rather than across the batch of images, so information between classes is not merged. Additionally, we included Attention layers (performs self-attention) to make the model understand which part of image is more important. When evaluating this updated model, we observed that the produced images more closely resembled the images in the training set.

Having a successful working model for unconditional generation, we pivoted to the task of conditional generation. The conditional model variants are just like our unconditional model with one key change - the inclusion of text encodings. At first, we incorporated tokenized text into the model by passing it through a sinusoidal embedding which would then be passed through a linear layer. After training the model on a reduced set of 30 images, now including tokenized captions, our model was able to produce coherent images. However when it came time to test the model, the caption we input to the trained model would not produce an image that was reflective of the provided caption - the model reproduced images from our training set seemingly at random. We figure the linear layer was not able to properly attend to the part of the text embedding that could produce the desired image.

As a result, we decided to incorporate cross-attention between the image and the text encodings to enable the model to learn a more useful representation of the caption. After applying cross-attention and training our model on the previously described smaller training set, we found the model was now able to produce images that reflected the input captions as shown in Figure 12.

After identifying our best architecture, we were limited to training the Unconditional and Conditional models on the full data set for only 96 epochs, which we believe does not showcase the true potential of either model. At 96 epochs, both models had plateaued loss values, but training for longer may help the models converge on a better solution. Due to the noise added by the text encodings, the conditional model most likely requires more training epochs.

4.3.2 Learning Rate

We tested a few learning rates between $1e^{-6}$ and $1e^{-2}$. We found that $1e^{-4}$ had the most stable and consistent learning. We saw that a slightly larger learning rate was performing better, so we increased the rate to $3e^{-4}$ to train our final model.

4.3.3 Miscellaneous

Initially, we implemented early stopping in an effort to save computation and prevent overfitting. We tried several settings for minimum loss deltas within a window of size 10. Loss deltas $\in [1, 10)$ ultimately stopped the model too early with noisy images. Removing early stopping and letting the model run for a fixed number of epochs allowed to reduce loss beyond local plateaus and ultimately produced better images.

We evaluated several activation functions for our models while varying the architectures. We started with ReLU after the convolutional filters in the downsampling and upsampling blocks. In an effort to improve performance, we tried SiLU and LeakyReLU but observed no performance improvement. When we added attention, we switched to GELU, which behaves similarly to SiLU but allows for more negative numbers after activation.

We experimented with a few values of rtol and atol within the ODE solver. The default values within the torchdiffeq package had the following configuration: integrator set to 'dopri5', atol = $1e^{-6}$, and rtol = $1e^{-9}$. However, when generating an image, the solver would take several seconds to output a single image. In response, we increased both atol and rtol to $1e^{-5}$, which lowered inference time to less than one second per image. Such a performance increase enabled us to perform FID evaluations on 1000 images in a reasonable amount of time. In order to ensure image quality was not lost between these configurations, we re-calculated FID scores on a batch of 1000 images and observed a mere difference of 3 points.

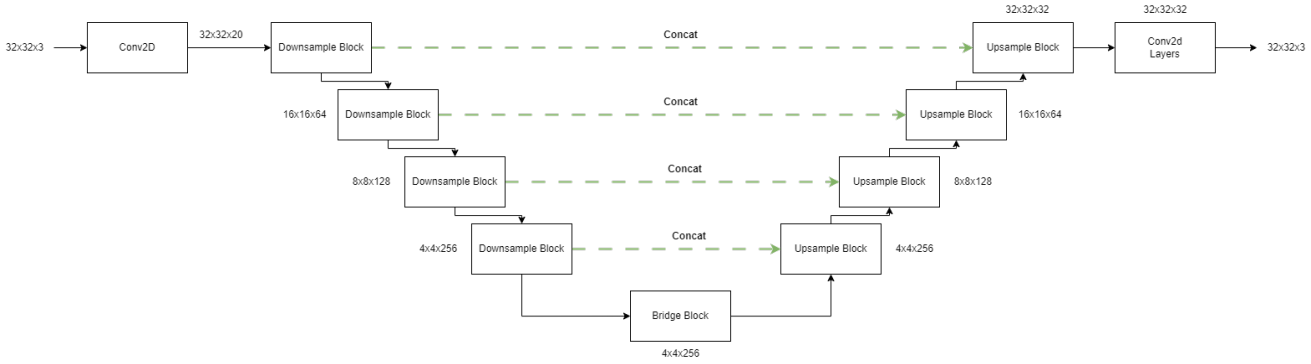
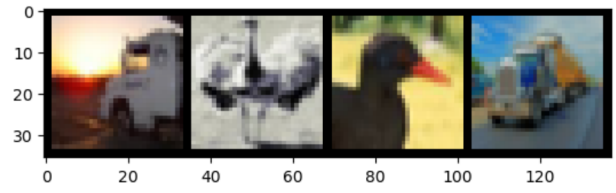


Figure 3. High-level architecture of UNet Model



```
truck bird  bird  truck
a white truck with a sunset behind it
emu - a bird with a wing spread
a black bird with red beak
a semi truck driving down the road
```

Figure 4. Sample Captions generated by BLIP2-Flan-T5. Line 1 displays the actual image labels, and at the bottom are corresponding generated image captions

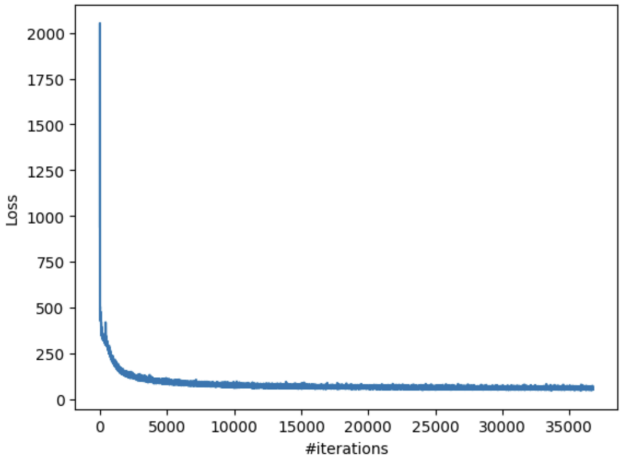


Figure 5. Loss curve of Unconditional Flow Matching Model

4.4. Evaluation Results

The lowest loss objective value for the unconditional and conditional models were 59.27 and 51.82 respectively (Loss curves during training are presented in Figure 5 and 6). To analyze the performance of Unconditional Flow Matching

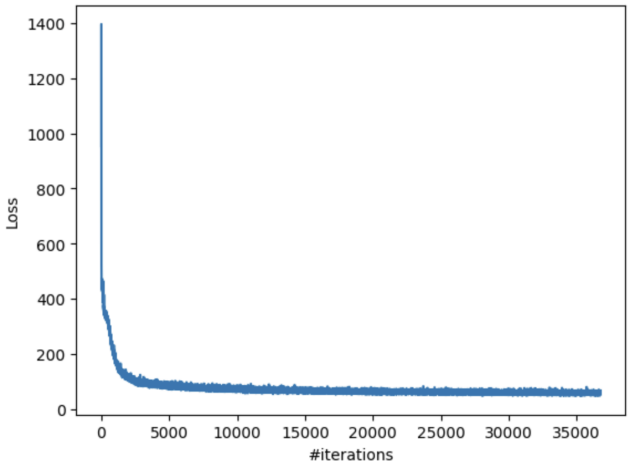


Figure 6. Loss curve of Conditional Flow Matching Model

Model, we calculated the FID score on 1,000 fake images generated using the model and the real images from CIFAR-10 test set. Similarly for conditional model, we generated 1000 fake images but no caption was provided as input.

Conditional generation capability of the model was then evaluated using CLIPScore, where 100 randomly sampled captions from the CIFAR-10 test set were provided as input to the model.

The evaluation results of the model's performance are tabulated in Table 1.

	FID (1K Images)	**CLIPScore (100 Images)**
Unconditional FM	105.54	-
Conditional FM	385.86	22.19

Table 1. Performance Evaluation of FM models

A set of two images demonstrating the effects of denoising a random image over 10 time steps with our unconditional Flow Matching Model is shown in Figure 7. As

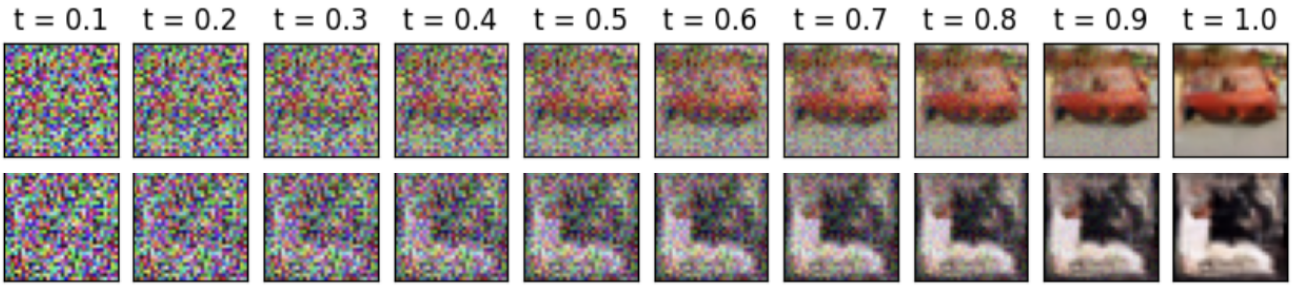


Figure 7. A sample of two images from our Unconditional Flow Matching Model, taken over ten time steps. The first image resembles a car and the second resembles a cat.

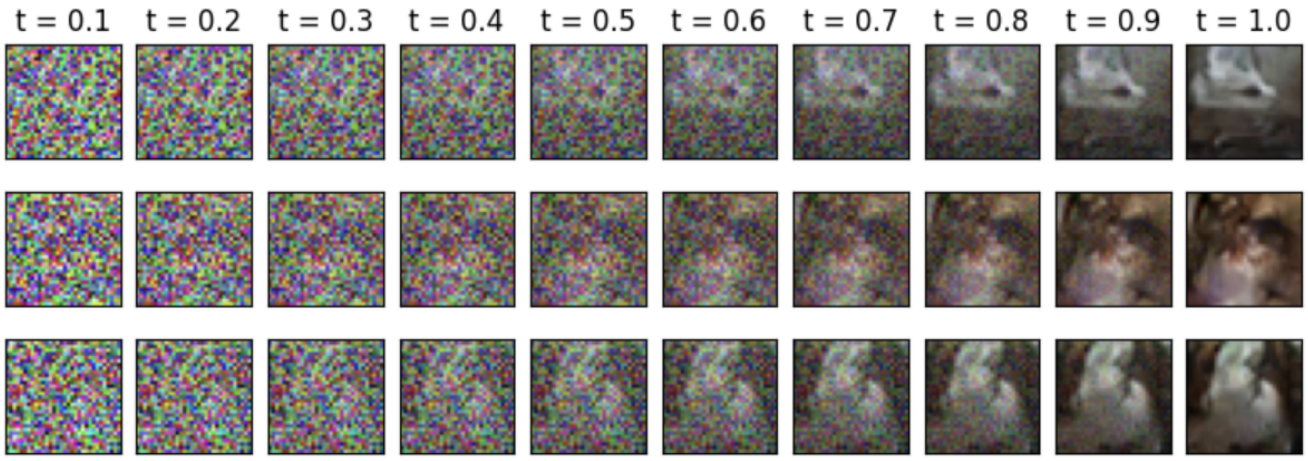


Figure 8. A sample of three images from our Conditional Flow Matching Model, taken over ten time steps. Captions: "a white cat on the floor", "a deer in a field", "a red car in a parking lot"

shown in the figure, our model is able to reproduce images (similar to that of the CIFAR dataset) such as a car and a cat in the first and second rows respectively.

We have presented the images generated by the conditional model on a set of provided captions in Figure 8. The model produced an image of a white cat that matches the description of the first caption, whereas not a clear coherence is observed between the generated images and provided captions for the rest.

5. Conclusion

Flow Matching has proven to be a stable and efficient alternative to the diffusion objective for unconditional image generation. In this project, we focused on evaluating and comparing the results of conditional and unconditional image generation utilizing the Flow Matching objective with Conditional Optimal Transport.

The experimental results demonstrate that conditional image generation using Conditional Flow Matching is feasible. We have shown that on smaller datasets the model is capable of generating detailed and coherent images that match the training set (See Figure 12). Moreover, the images are reproducible using the same input text, which shows that the model is conditioning the image generation on the text input.

When training on the larger datasets, both the unconditional and conditional models successfully generate images that have a natural composition. But upon closer inspection, we observe that the images produced by the conditional FM model are frequently inconsistent with the provided captions as demonstrated in Figure 8. We believe the reason for this behavior could be insufficient training time. Considering the variety present in the CIFAR-10 dataset, paired with the small number of training epochs, we believe with more iterations the models could have learned flows that produced higher-quality images.

6. Future Work

A few key areas we are keen to explore in the future that would help our model perform better are as follows. Firstly, increasing the training time for better coherence of the conditional model with the captions. Another improvement

could be the use of higher-resolution images, so that the model is able to process more information, and performance is more obvious to the human eye. Finally, we theorize that having a conditional image set with better descriptive phrases, would really help increase the model performance (Currently, the phrases are all pretty small, and some are very inaccurate).

References

[1] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations, 2019. 1

[2] Darius Chira, Ilian Haralampiev, Ole Winther, Andrea Dittadi, and Valentin Liévin. Image super-resolution with deep variational autoencoders, 2022. 1

[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. 1

[4] William Harvey, Saeid Naderiparizi, and Frank Wood. Conditional image generation by conditioning variational auto-encoders, 2022. 1

[5] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning, 2022. 2

[6] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018. 2

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. 1

[8] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018. 1

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019. 1

[10] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023. 1

[11] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets, 2014. 1

[12] NLP Connect. vit-gpt2-image-captioning (revision 0e334c7), 2022. 4

[13] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021. 1

[14] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2016. 1

[15] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. 2

[16] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 1

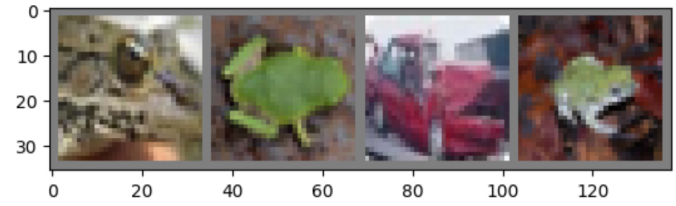
[17] Salesforce. blip2-flan-t5-xl, 2022. 4

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. 2

[19] Lilian Weng. Flow-based deep generative models. *lilianweng.github.io*, 2018. 1

[20] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion models in generative ai: A survey, 2023. 1

A. Appendix



```
frog  frog  car   frog
torch.Size([4, 3, 32, 32])
<class 'PIL.Image.Image'>
<class 'PIL.Image.Image'>
<class 'PIL.Image.Image'>
<class 'PIL.Image.Image'>
['an old photo of a person holding a bunch of colorful feathers',
 'a colorful kite flying through the air',
 'a cake with a picture of a train on it',
 'a colorful kite is hanging from the side of a tree']
```

Figure 9. Sample Captions generated by VIT GPT2. Line 1 displays the actual image labels, and at the bottom are corresponding generated image captions

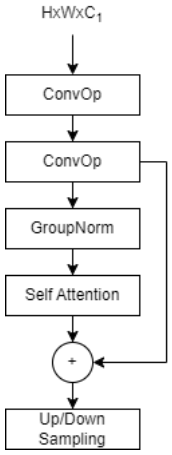


Figure 10. ConvNet Block Design

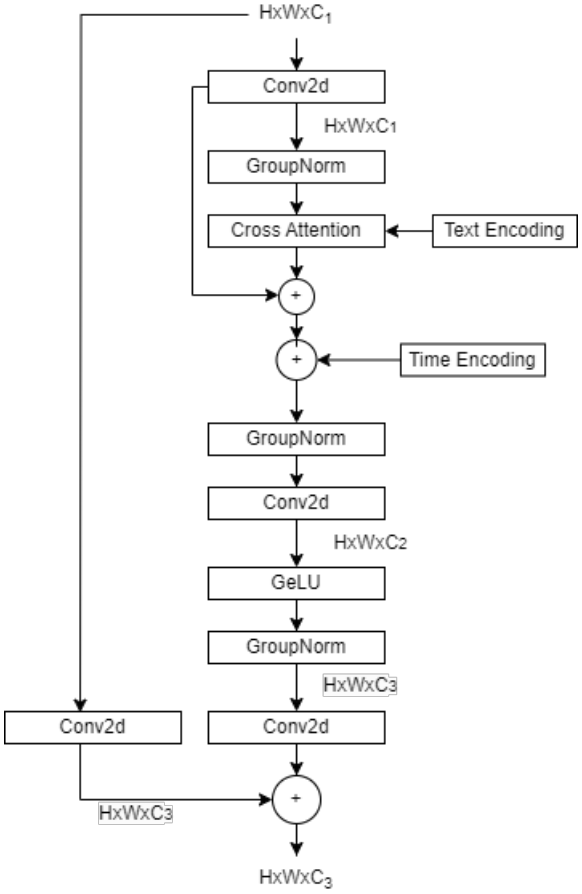


Figure 11. Operations performed inside ConvOp block

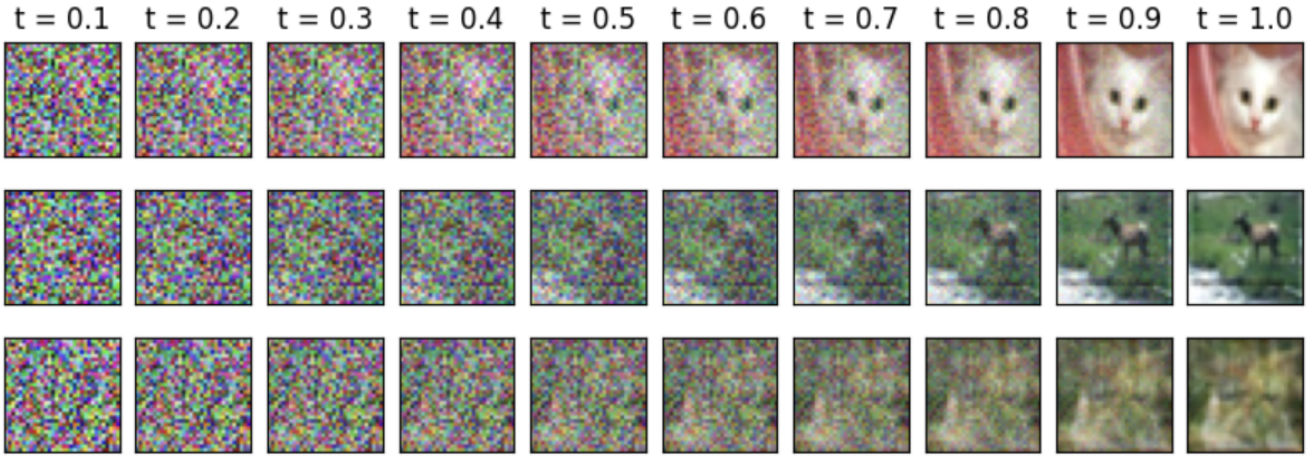


Figure 12. A sample of three images from our Conditional Flow Matching Model, taken over ten time steps. Trained on a small data set of 30 images. Captions: ”a white cat peeking out of a pink curtain”, ”the elk is standing in the grass”, ”the cat is black with yellow eyes is standing in the field”