Lab report

Group MikhelsonPallotta

Mikhelson German

September 30, 2024

VDT: GENERAL-PURPOSE VIDEO DIFFUSION TRANSFORMERS VIA MASK MODELING

of Prof. Dr. Juergen Gall

Fall Semester 2024

Tutor:

Enrico Pallotta

Prof. Dr. Juergen Gall

Contents

Abstract

This report outlines the tasks performed in implementing and training the Video Diffusion Transformer (VDT) model for video prediction and unconditional generation tasks using the CityScapes dataset. The primary objectives included implementing data preprocessing, metric evaluation, training/validation functions, and reproducing experimental results. The results, challenges, and suggestions for future improvements are discussed. Code can be found at: `https://github.com/Wektor607/VDT`

1 Introduction

In this paper (1), the authors provided code with the model architecture, an example of inference code, and an implementation of the *Denoising Diffusion Probabilistic Model* (DDPM). The following tasks were set for me:

- Implementation of data loading and processing.
- Implementation of metrics.
- Implementation of the training and validation functions.
- Reproducing the experimental results for *Video Prediction* and *Unconditional Generation* tasks using the *CityScapes* dataset.

2 Video Diffusion Transformer

The Video Diffusion Transformer (VDT) is a novel approach that leverages transformer-based diffusion techniques to generate high-quality video frames. Unlike existing methods that often struggle with effectively modeling temporal information and handling diverse video generation tasks, VDT excels by integrating transformer blocks with temporal and spatial attention modules. These blocks enable the model to capture long-term dependencies and dynamic object movements over time, ensuring high-quality and consistent video frames.

VDT utilizes a pre-trained **Variational Autoencoder** (VAE) tokenizer to project video clips into a latent space, reducing computational complexity. The latent features are divided into non-overlapping patches, and spatial-temporal **Sine-Cosine** positional embeddings are added to learn spatial and temporal information. The approach's adaptability is further enhanced by its ability to handle various video generation tasks such as

- Unconditional Generation
- Video Prediction
- Video Backward Prediction
- Image-to-Video Generation
- Arbitrary Video Interpolation
- Spatial-Temporal Video Completion

This flexibility makes VDT a robust and scalable solution for video generation challenges.

3 Implementation

The implementation process consisted of the following:

- Developed a data preprocessing pipeline *FrameDataset*
  - Each video sequence in this dataset consists of 30 frames
  - Every video is converted into a list of 30 individual frames
  - During **training** and **validation**, the model was trained on 16 frames. A random start and end index were selected to define a 16-frame range, which was then extracted from the complete sequence of 30 frames
  - For **inference**, the model was evaluated on the complete video sequences
- Developed a *MetricCalculator* class for calculating metrics such as
  - Structural Similarity Index Measure (**SSIM**)
  - Learned Perceptual Image Patch Similarity (**LPIPS**)
  - Peak Signal-to-Noise Ratio (**PSNR**)
  - and Fréchet Video Distance (**FVD**)
  - The system is flexible for adding new metrics
- Implemented the training, validation, and test functions.
  - The training function employs the *CosineAnnealingLR* learning rate scheduler.
  - The *Denoising Diffusion Probabilistic Model* (**DDPM**) serves as the core diffusion model.
  - The training setup is optimized with *Distributed Data Parallel* (**DDP**) to leverage multiple GPUs, ensuring efficient training on large-scale data.
  - The model supports both training and inference modes, making it adaptable for different tasks.

4 Dataset and Training/Inference Configuration

For the experiments, the *CityScapes* (2) dataset was used, which contains video sequences captured in German urban environments. Each video consists of 30 frames of dynamic street scenes.

The following tasks were focused on:

- **Video Prediction**
  As by paper description, the model training was conducted using a sequence of **16 frames**, consisting of:
  - **8 conditional frames** provided as input

  - **8 frames** predicted by the model

  During the inference stage, the model was evaluated on a **full 30-frame sequence**, where:

  - The **initial 2 frames** were used as conditional inputs
  - The **remaining 28 frames** were generated by the model

- **Unconditional generation**
  All frames were masked, and the model had to generate the video frames from scratch without any initial condition frames.

5 Challenges and Setup

The following **challenges** arose during this project:

- There was no detailed information on the **computational resources** used in the original paper for training the model, including how much time and what kind of GPUs were needed.

- **Some important training parameters** were not provided, such as the sampling steps used during training and inference, number of epochs, which made it challenging to reproduce the exact conditions described in the paper.

- During training and testing, I occasionally encountered **issues with the FVD metric**. This problem arose when the dataset was split into batches, as the last batch could sometimes be smaller than the others, leading to dimension mismatch errors. To address this issue, I opted to ignore such incomplete batches, ensuring consistent batch sizes throughout the process.

- The model was trained on the Bender supercomputer. Due to the **high demand for the A100 GPUs**, my jobs were automatically assigned the lowest priority, resulting in a wait time of 1-2 days in the queue. Consequently, I decided to continue training on the A40 GPU, which required reducing the batch size since the A40 has near half the memory capacity of the A100. However, the repository contains both versions of the model before using A40 and after.

- Experiments were conducted using a **smaller version of the model**. While this version allowed for faster training, the significantly reduced number of parameters (approximately 20 times fewer) had a detrimental impact on the overall accuracy. As a result, the decision was made to proceed exclusively with the larger model version to maintain performance.

- Considerable effort was invested in tuning the **diffusion number of steps**, with values carefully selected from a range of [16, 1000].

- Initially, it was assumed that the model would be trained to perform **6 different tasks**, as described in the original article. For each iteration of training, a task was randomly selected, an appropriate mask was generated, and the model began training. However, this approach had a negative impact on the final results. Consequently, the number of task types has been reduced from **6 to 2** to simplify model training.

The **setup** used in this implementation included:

- The **VDT-L/2** model.
- 5 Nvidia **A100** GPUs (80GB each) and 4 Nvidida **A40** GPUs (48GB each) for training.
- Training model:
  - **979 epochs** on 5×A100
  - **1124 epochs**: 406 epochs on 5×A100 + 718 epochs on 4×A40
    - **Remark:** While continuing to train the model on the A40, I was unable to train it on the A100 immediately after submitting the job via SLURM. As a result, the total number of training epochs on the A100 differs from the epochs trained exclusively on the A100.
- **AdamW optimizer** with a **learning rate** of 1×10^{-4} and **weight decay** of 0.0.
- **Batch size**
  - On A100: **16**
  - On A40: **6**
- **Frame size** of 128×128
- **Diffusion sampling steps** of 500 during training and inference
- **Loss function**: MSE

6 Results

The results are presented in two categories: Video Prediction and Unconditional Generation.

First, consider the problem of predicting a video with 30 frames, where it is necessary to predict 28 frames. The orange highlights the given frames. As you can see, on Figure 1, literally in the 3rd frame on the left, buildings turn into trees, the markings turn into solid lines, and the car is slightly deformed. The surroundings and objects become even more blurred.

The same trend can be seen in the Unconditional Generation tasks. Figure 2

The low quality of these results is likely due to the model not being fully trained. As mentioned earlier, the authors did not specify the duration of the training process. Another potential issue is the number of steps in the diffusion model. Increasing the number of steps can improve the quality of the generated samples, but it also significantly slows down training. Instead of using DDPM, it is recommended to try the DDIM sampling algorithm, which can generate higher-quality samples more quickly and with fewer steps. Additionally, increasing the batch size can lead to more stable results and better overall quality. However, the main limitation is the available GPU memory.

Table 1 presents the results for Video Prediction tasks. While VDT1124 (trained on two different types of GPUs) showed slightly better performance than VDT979 based on the FVD metric, the improvement is minimal, especially considering that VDT1124 was

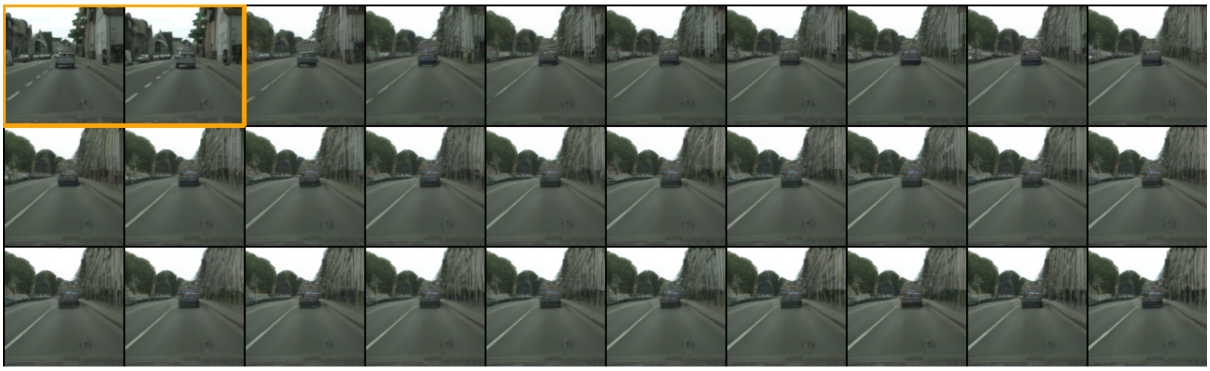

Figure 1: Video prediction task. VDT979

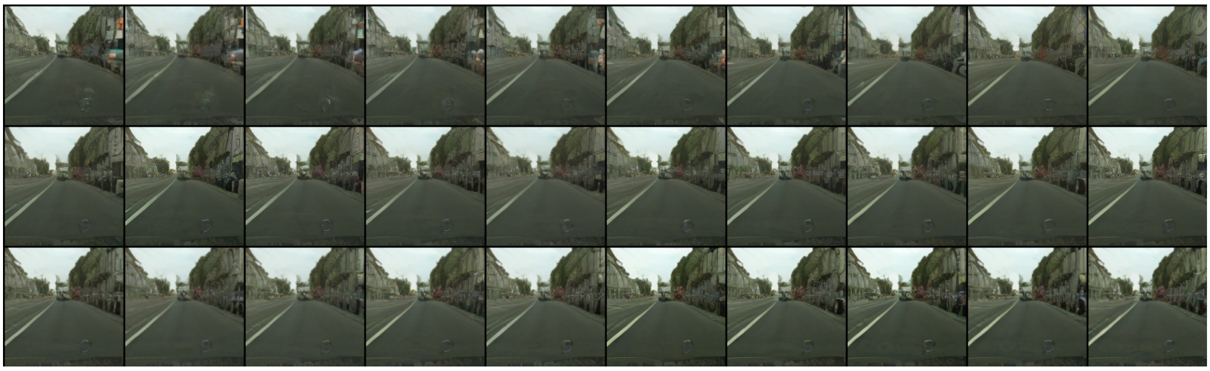

Figure 2: Unconditional Generation task. VDT979

trained for over 100 more epochs. This discrepancy is largely due to VDT1124 being fine-tuned with a smaller batch size, a consequence of memory limitations on the A100 GPU. Despite these differences, both versions of the model still lag behind all other models in terms of quality and require further optimization during training.

7 Conclusion

- The suboptimal quality of the results can be attributed to the model not being fully trained, as the authors did not specify the exact training duration.
- Additionally, the number of diffusion steps plays a significant role in the model's performance. While increasing the number of steps generally leads to improved results, it also results in a longer training time.
- To enhance performance, we recommend experimenting with Denoising Diffusion Implicit Models (DDIM), which are capable of generating high-quality samples similar to DDPM but with faster generation times and fewer diffusion steps.
- Due to limited access to A100 GPUs, part of the training had to be conducted on A40 GPUs, which have significantly less memory. This constraint necessitated reducing the batch size from 16 to 6. Since larger batch sizes typically result in more stable and consistent training outcomes, this reduction contributed to the slight performance gap between VDT1124 and VDT979.

Table 1: Video prediction on Cityscapes (128 × 128) conditioning on 2 and predicting 28 frames.

Cityscapes	**FVD↓**	**SSIM↑**
SVG-LP	1300.3	0.574
vRNN 1L	682.1	0.609
Hier-vRNN	567.5	0.628
GHVAE	418.0	0.740
MCVD-spatin	184.8	0.720
MCVD-concat	141.4	0.690
VDT	**142.3**	**0.880**
VDT979	1607.1	**0.406**
VDT1124	**1555.7**	0.402

References

[1] Lu, H., Yang, G., Fei, N., Huo, Y., Lu, Z., Luo, P., Ding, M. (2023). Vdt: General-purpose video diffusion transformers via mask modeling. arXiv preprint arXiv:2305.13311: https://arxiv.org/pdf/2305.13311

[2] CityScapes Dataset: https://www.cityscapes-dataset.com/