Generative Video Augmentation for Image Classification

Robert Ashe, Afnan Algharbi, Thiago Henriques, Adam Lizerbram, Bieu To

Abstract

Image classifier models traditionally rely on single frames, yet humans routinely exploit motion cues to categorize and distinguish visually similar objects. We investigate whether introducing a synthetic temporal dimension into static 2-D image datasets can improve the accuracy of image classifiers, and what impact this has on the computational performance of these models. Starting with a conventional VGG-11 backbone fine-tuned on a 10-class subset of Tiny ImageNet, we apply Stable Video Diffusion (SVD) to hallucinate two additional frames for every image in our dataset, yielding short 3-frame videos that mimic subtle object motion. Pre-processing the 5000 training images and 500 validation images in our dataset using SVD on a single RTX-3060 GPU required 29 hours, resulting in a 3-D augmentation of our original dataset. An "inflation" trick is used to convert each 2-D convolution, batch-normalization, and pooling layer in VGG-11 to its 3-D counterpart, allowing the network to learn spatial and temporal features while reusing the original pretrained weights to initialize the 3-D VGG-11 before fine-tuning on our 3-frame video dataset. Compared to the 2-D baseline, the proposed SVD-augmented model improves Top-1 accuracy from 54.2% to 75.8% and Top-3 accuracy from 77.8% to 90.4%, while incurring a modest 0.14 ms increase in average inference time per sample. Per-class analysis and confusion matrix heatmaps reveal improvements for visually similar classes and reduced class-confusion overall, confirming that the synthetic temporal information provides increased context rather than noise. The study demonstrates that Stable Video Diffusion can serve as a practical data-augmentation engine, which is not conventionally explored in the domain. Our findings highlight the potential for exploiting generative video models to enhance static datasets when genuine video data is unavailable.

1. Introduction

Traditional image classification models trained on static images often lack the temporal context which could improve accuracy. In this work, we explore the following question: can image classification performance be improved by artificially introducing a temporal dimension to static data? We use Stable Video Diffusion (SVD) to generate two additional frames per image to form short pseudo-video clips. These augmented clips are used to train an inflated version of a VGG-11 classifier model, where each 2-D layer is substituted with its 3-D counterpart. Both models are evaluated, with the baseline VGG-11 fine-tuned on a 10-class subset of the Tiny ImageNet dataset compared against the inflated VGG-11 that is fine-tuned on the same dataset augmented using SVD. We find that the inflated model achieves about 20% higher Top-1 accuracy while introducing a minimal increase in inference time.

To our knowledge, this particular application of SVD specifically to enhance image classification through data augmentation has not been explored before. Our key contributions are:

1. Introduce the application of SVD to synthetically augment static images to generate a pseudo-temporal dimension for the purpose of image classification.

2. Compare common performance metrics of each model to determine feasibility of using generative video models for data augmentation to improve existing classifiers.

2. Related Works

A foundational work in motion synthesis is "Everybody Dance Now" by Chan et al. [3], which introduced a pose-based video-to-video translation framework for motion transfer. In contrast to their work, which requires a source video and explicit pose conditioning, our project starts from a single static image and generates additional frames using SVD to simulate motion. The end goal differs as well: instead of generating photorealistic video content, we use synthetic motion to enhance image classification performance by augmenting static image data using a generative video model (SVD).

More recently, Casarin et al. [2] proposed "Your Image is My Video: Reshaping the Receptive Field via Image-To-Video Differentiable AutoAugmentation and Fusion" that has a similar image-to-video modality as our project. However, they use Differentiable AutoAugmentation (DAS) instead of SVD to leverage temporal context to improve im-

age classification. The drawbacks of this approach include high memory usage, general limitations across tasks and architectures, dependence on specific transformation searches, reduced robustness to temporal shuffling, and a lack of evaluation on transformer backbones.

Fu et al. (2024) contributed with "GenDDS: Generating Diverse Driving Video Scenarios with Prompt-to-Video Generative Model", using the "Stable Diffusion XL fine-tuned with Low-Rank Adaptation (LoRA)" by introducing a diffusion-based system similar to SVD. The shortcomings of this approach indicated that the model only relies on a single dataset, lacks quantitative evaluation, does not handle the increase of video length and temporal complexity, and uses manual review of auto-tagged frames.

In summary, while previous works demonstrate the power of generating or transferring motion for content creation, simulation, or augmentation, our contribution quantifies the impact of synthetic motion on recognition tasks, using a clean 2-D vs. 3-D comparison. This bridges the gap between motion transfer, generative video synthesis, and practical classification performance, especially in low-resource settings where real video data is scarce.

3. Methodologies

3.1. Sourcing Data

We start from the official Tiny-ImageNet dataset, an open source dataset 2-D static 64 x 64 RGB images. The dataset originally contains 200 classes with each class containing 500 training and 50 validation and test images each per class, totaling a sum of 100,000 training images and 10,000 validation images. We curate 10 semantically related mammal categories.

The original dataset contained all training images in subdirectories based on the class's sysnet (unique identifier associated with each class label in the Tiny ImageNet dataset), so they were relatively simple to be picked out. However, the validation images coexisted for all classes in one folder. The Tiny Imagenet dataset provided an annotation file that mapped each validation sample via its synset ID to its respective class. Through several processing scripts, all validation images were extracted and sorted into each of the 10 subclasses and organized in their own subdirectories.

The 10 class subset of the Tiny ImageNet dataset was chosen to limit GPU memory usage and accelerate video generation. The resulting subset contains 5,000 training, and 500 validation images in total. The 10 classes selected for our dataset are shown in **Table 1**.

Synset	Class
n02099601	Golden Retriever
n02123394	Persian Cat
n02129165	Lion
n02132136	Brown Bear
n02403003	Ox
n02415577	Bighorn Sheep
n02423022	Gazelle
n02481823	Chimp
n02504458	Elephant
n02509815	Red Panda

Table 1. Selected classes and their corresponding synsets.

3.2. Baseline Model

The baseline model used in this project is a standard VGG-11 convolutional neural network (CNN). The main purpose of using this architecture is attributed for its wide recognition, effective nature, and simplicity in application [5], especially for the relatively small size of the dataset. The network was fine-tuned, and the classifier layer was adjusted to manage the 10-class subsets of the Tiny ImageNet dataset. With it already being pretrained on ImageNet, the use of this VGG-11 model allows a fair outlook that any enhancement in the classification performance can be more credited to the data augmentation rather than the model itself.

The finetuned 2-D baseline model is crucial in our implementation as the predicted labels of the images yield a benchmark for assessing the accuracy of the classification task for eventual comparisons. This tactic provokes a reference point for evaluating the introduction of synthetic video augmentation for future steps in this experiment. Figure 1 demonstrates the execution path for training the baseline 2-D model, which is a primary step for our approach.

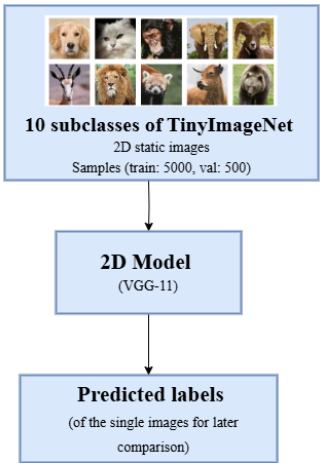

Figure 1. Pipeline for the Baseline 2-D Classification model

3.3. Stable Video Diffusion (SVD)

This stage in the procedure plays a prominent role in achieving an understanding of the influence that synthetically prepared temporal progression has on image classification tasks, conveying the core purpose behind this study [4]. To examine this prospect, we introduced augmentation to each image in the 10-class subset using a Stable Video Diffusion Model (SVD). Precisely, the employed SVD model used is a variation of the img2vid model provided on Hugging Face, an open-source interface that hosts a variety of state-of-the-art models. This SVD model is designed to generate consistent synthesized sequences of video frames derived from a single image as the input. Using stable diffusion is ideal in constructing simulated temporal context when real video data is scarce. The general structure of SVD executes as follows:

1) Encoding the sample: Given a 2-D-static image, the SVD starts by encoding it with a pretrained encoder built in the actual stable diffusion framework much like a Variational autoencoder(VAE) to introduce latent representation in the sample's space. This process prepares the sample for the introduction of temporal context.

2) Temporal Conditioning: For this step, noise is added into the latent space to simulate a corrupted like form of a short video clip. This is when the sequence of synthesized frames is generated.

3) Denoising: The model then iteratively removes the added noise through a number of inference steps. The SVD model is trained to refine the sample to a more coherent video space based on the patterns recognized during the learning process of motion and temporal integration.

4) Decoding: The denoising process is followed by converting the latent representation of the sample back to pixelated RGB images through a video decoder which results in the production of a video clip.

3.3.1 Synthetic Video Augmentation Process

We start by preprocessing the data for the inflated 3-D model. Each sample in the 10 subclasses was upscaled to 224 x 224 pixels to accommodate the special dimension specifications of the SVD model's architecture. We used the diffusers library to leverage the Stable Video Diffusion pipeline and set a constant number of frames which is applied on the upscaled dataset, stacking the frames in a temporal dimension. For every still image, we generate two additional frames using SVD, producing 3-frame, 8-bit RGB clips.

During the diffusion process, each sample underwent encoding to latent space, denoising through 12 inference steps, temporal conditioning, then decoding back into a pixel image space. This process resulted in tensors of the shape, [T, H, W, C] in which C represents the RGB channels, T is the number of frames, H is the height of the sample, and W is the width. We retain the original 64 x 64 spatial resolution for fair comparison with the baseline model. The pretrained SVD model invoked with 12 diffusion steps, required approximately 29 hours of processing to augment the entire Tiny ImageNet subset using a single NVIDIA RTX 3060 (12 GB) GPU.

During the image augmentation process, the label from each original static image was recorded in a CSV file along with an index that together mirrors the data topology of the original dataset. The result of the synthetic video augmentation process was a 3-D analogue of 6fps (to allow easy viewing for human interpretations) MP4 videos of the subset from the 2-D Tiny ImageNet dataset. **Figure 2** exhibits the pipeline used during the augmentation process of the static samples via SVD, generating a dataset of synthesized video clips ready for the subsequent stage.

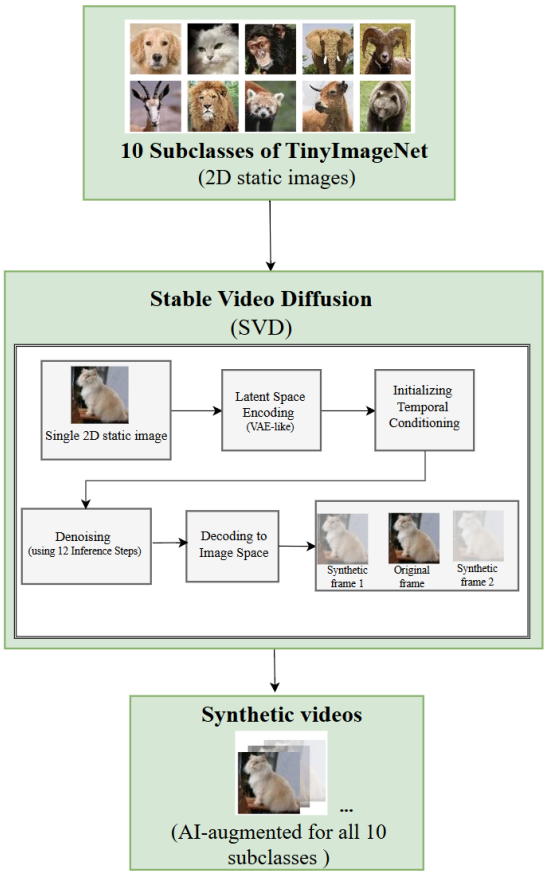

Figure 2. Preprocessing the synthetic frames using SVD.

3.4. 3-D Model

Having an input with a temporal dimension required the adaptation of the baseline 2-D model to transform into its 3-D counterpart with the goal of accommodating the extra dimension. To address this, we inflated the pretrained VGG-11 model into its 3-D equivalent by converting the 2-D convolutional layers of the model to 3-D. Furthermore, we introduced the addition of a temporal kernel initialized with a central time slice to preserve the pretrained weights of the 2-D VGG-11 model. The conversion was also applied on the 2-D Batch normalization and pooling layers to adapt to their 3-D form. The result was an inflated VGG-11 3-D model equipped to train on the augmented video data generated by the SVD process in the previous step. Training this architecture involved the use of a custom data loader that loads each augmented clip and their corresponding labels from the dataset. We used cross-entropy loss, Stochastic Gradient Descent (SGD) optimization, and learning rate scheduling to assess the model's performance. The outcome involved the production of the model's predictions of classifying the video labels for each sample, ready for comparison with the outputs generated from the classification of the Baseline model. **Figure 3** illustrates the training process for the inflated VGG-11 model that takes the augmented samples as input.

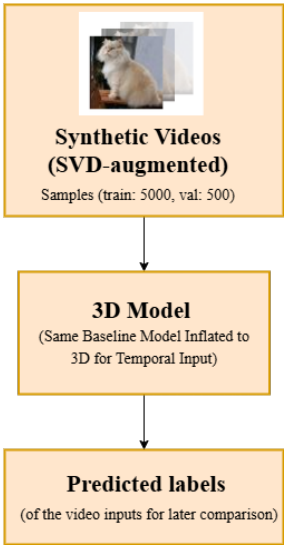

Figure 3. Pipeline for the 3-D classification model.

4. Experimental Results

4.1. Training and Validation Splits

To start the experimental analysis, proper data splitting was fundamental in validating the model's learning abilities. The spatial-only baseline and our inflated 3-D model are both trained on 5,000 training examples and validated on the 500 validation image split, and all model performance comparison data is based on the results of inference on the validation set.

4.2. Model Configurations

4.2.1 2-D Baseline

As mentioned earlier, we finetune the Torchvision VGG-11 for 20 epochs at 224 x 224 input resolution using SGD, LR = 1e-3, momentum = 0.9, weight decay = 1e-4, and batch size = 32. The classifier head is replaced with a 10-way fully-connected layer and randomly re-initialized before training.

4.2.2 Inflated 3-D VGG-11

Every 2-D convolution, batch-norn, and max-pool layer is converted to its 3-D analogue by replicating kernel weights along the temporal axis (i.e. k x k $\rightarrow$ 1 x k x k, then learn the temporal extents). We initialize from the 2-D VGG-11 baseline, set the temporal kernel size to 3, temporal stride to 1, and keep the same spatial hyperparameters from the finetuning of the 2-D baseline. 3-D VGG-11 training mirrors the 2-D process, except that batch size is now 8 (due to the threefold increase in memory).

4.3. Evaluation Metrics

To quantify our results, we report Top-1 and Top-3 accuracy over the 10-class validation set; per-class precision, recall, and F1; confusion matrices; and average per-sample inference latency (using PyTorch 2.7, CUDA 12.4, measured over the full 500-example validation set). **Table 2** shows the classification accuracy of the 2-D VGG-11 and the 3-D VGG-11 that was trained on our augmented dataset. Notably, we measured a 21.6% increase in Top-1 accuracy in the 3-D model, a strong indicator that the SVD data augmentation provided the model more context to make its predictions.

In **Table 3**, evaluation metrics related to model training are shown. Both models were trained with identical hyperparameters, with the exception of batch size due to the increased memory footprint of the 3-frame video clips in the augmented dataset. As expected, large jumps in model complexity were observed between the 2-D baseline and the inflated 3-D model. Training time on an RTX 3060 increased from 18 minutes to 192 minutes, the number of parameters increased from 128.8M to 147.2M, and the MACs (multiply-accumulate, 1 MAC $\approx$ 2 FLOPs) exploded from 7.65 GMac to 67.64 GMac in the 3-D VGG-11 model.

Model	Input	Top-1 (%)	Top-3 (%)
2-D VGG-11	Single frame	54.2%	77.8%
3-D VGG-11	Aug. frames (SVD)	75.8%	90.4%

Table 2. Classification accuracy of 2-D vs. 3-D VGG-11.

Item	2-D VGG-11	3-D VGG-11
GPU	RTX 3060 12GB	RTX 3060 12GB
Epochs	30	30
Batch Size	32	8
Training Time	18 min	3 h 12 min
# Parameters	128.8M	147.2M
MACs	7.65 GMac	67.64 GMac

Table 3. Training implementation details for both models.

4.4. Per-Class Analysis

Figure 4 shows the precision, recall, and F1 score comparison for each class in our dataset. Averages for each of these metrics improved significantly from the 2-D model to the 3-D model: precision increased from 0.62 to 0.75 (+0.13), recall increased from 0.54 to 0.75 (+0.21), and F1 improved from 0.53 to 0.75 (+0.22). The largest gains occurred for *red panda* (+0.31 F1) and *brown bear* (+0.27 F1), two visually similar classes frequently confused by the 2-D model. Figure 4 summarizes a comparative plot for the precision, recall, and F1 scores for each model's classification performance per the ten subclasses.

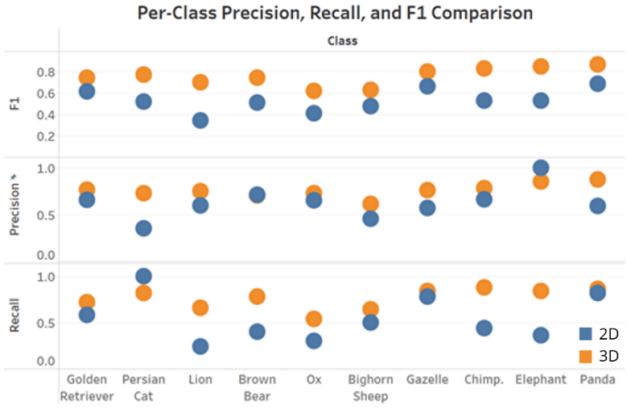

Figure 4. Per-class precision, recall, and F1 score comparison.

4.5. Confusion Reduction

Figure 5 and **Figure 6** visualize the confusion matrices for the two models. Off-diagonal mass (misclassifications) drops 42% for the video model. Particularly troublesome classes like the *Persian cat* and the *red panda* that were frequently confused by the 2-D model (see vertical artifacts in 2-D VGG-11 matrix) were predicted much more accurately in the 3-D model. Notably, *red panda* ↔ *brown bear* confusions fall from 8 to 0 instances, eliminating the challenges with ambiguity between these visually similar classes.

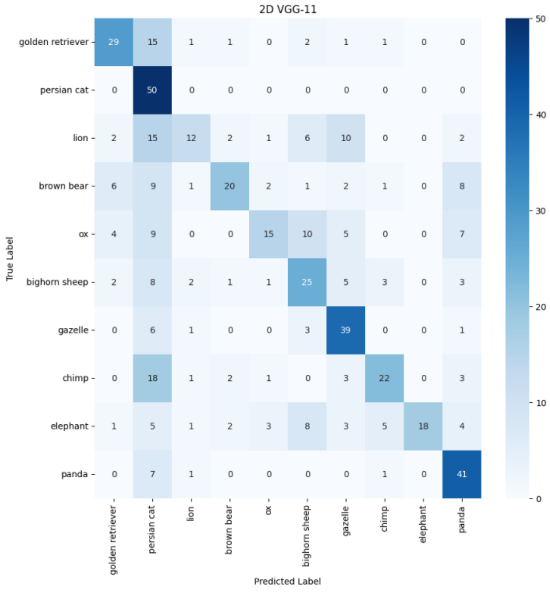

Figure 5. 2-D model confusion matrix

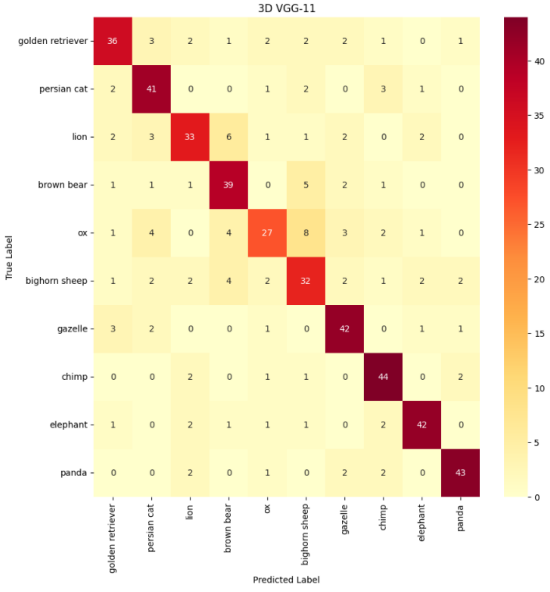

Figure 6. 3-D model confusion matrix

4.6. Computation & Efficiency

Inference latency increased by an average of 0.00014s (0.14 ms) per sample (0.000375 $\rightarrow$ 0.000513s), a 36.8% expansion in computational overhead that is negligible for offline prediction and acceptable for many real-time workloads, especially those that prioritize accuracy over efficiency.

5. Limitations and Future Work

Our experimental setup isolates the learning benefit of synthetic motion by preprocessing every static image with SVD prior to the training and evaluation process for the inflated 3-D VGG-11 model. While this decoupled design simplifies analysis, it prevents us from assessing the true computational overhead of a production system that would:

1. Receive a single RGB image at inference time,
2. Invoke SVD on-the-fly to hallucinate additional frames, and
3. Feed the resulting video clip directly into the 3-D classifier for prediction.

Consequently, our reported latency figures in Section 4 exclude both the SVD generation cost and any data movement between modules in this proposed pipeline.

Future work on this project could include building this fully automated 2-D image $\rightarrow$ SVD frame generation $\rightarrow$ 3-D inference pipeline, which would allow us to benchmark end-to-end runtime of this system to thoroughly quantify the potential benefits of this architecture for real-world applications. Establishing this pipeline will provide a realistic example for other researchers and practitioners that wish to adopt pseudo-temporal data augmentation in latency-sensitive applications.

6. Conclusion

Based on the results from our experiments, there are three reasons that explain the improvements in the model's performance. Firstly, the addition of pseudo-temporal context allowed the model to recognize how features changed over time, which resulted in an improved ability to detect patterns, especially between visually similar classes, such as the *brown bear* and *red panda*. Secondly, by inflating the model to a 3-D architecture to make it work with short videos, this allows the model to extract features across both spatial and temporal axes while maintaining the integrity of the original VGG-11 structure. Finally, by introducing additional synthetic frames for each sample in the training data, the model was trained with a larger amount of information, thus reducing the chance of overfitting. Being able to observe how a target changed over time allowed the 3-D model to have a more robust understanding of necessary features which resulted in better decision-making. Overall, this experiment illustrates how generative AI models can help improve the quality of training data.

In conclusion, despite the 36.8% increase in the calculation time for each sample, the remarkable rise of the Top-1 accuracy from 54.2% to 75.8% proved the tradeoff to be worthwhile. The experiment of using Stable Video Diffusion to generate synthetic data for model training proved that generative AI can also be used for enhancing data quality, not just for content generation. This opens another path for improving the performance of machine learning models by augmenting training data with generative AI. The future work of this can involve extending the number of frames to generate and testing with different models for data generation.

7. Individual Contributions

Afnan Algharbi curated the data used for model training and evaluation, wrote the Methodologies section in the final report, organized project repositories and handled document formatting. Robert Ashe designed and implemented the baseline model, SVD data augmentation, and inflated model architectures. He also wrote the Experimental Results and Limitations and Future Work sections. Thiago Henriques researched relevant papers, and wrote the Related Works section. Adam Lizerbram created diagrams, structured the final report, and wrote the Abstract and Introduction sections. Bieu To analyzed the results and wrote the Conclusion section.

References

[1] Blattman, Andreas et al. ”Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets” (2023) https://arxiv.org/abs/2311.15127

[2] Casarin, Sofia, et al. “Your Image is My Video: Reshaping the Receptive Field via Image-To-Video Differentiable AutoAugmentation and Fusion” (2024) https://arxiv.org/abs/2403.15194 1

[3] Chan, Caroline, et al. “Everybody Dance Now.” (2019). https://arxiv.org/abs/1808.07371 1

[4] Fu, Yongjie, et al. “GenDDS: Generating Diverse Driving Video Scenarios with Prompt-to-Video Generative Model” (2024) https://arxiv.org/pdf/2408.15868 3

[5] Simoyan, Karen and Zisserman, Andrew ”Very Deep Convolutional Networks for Large-Scale Image Recognition” (2015) https://arxiv.org/abs/1409.1556 2