Improving Image Classification through Diffusion-based Dataset Augmentation

Derek Paulsen [*1] **Bojun Liu** [*1] **Gabriel Gozum** [*1] **Joe Gorka** [*2]

Abstract

Image classification task has been widely investigated in machine learning field, however, its accuracy is usually limited by the amount of accessible labeled training data. In recent years, diffusion-based text-to-image generative models have been rapidly developed and exhibited great success on generating novel visual artworks with high efficiency and flexibility under the guidance of given text. In this work, we propose the diffusion-based dataset augmentation method, which employs the stable diffusion model to generate synthetic data for dataset augmentation, and successfully improve the accuracy of the image classification models.

1. Introduction

1.1. Motivation

In recent years deep learning models have achieved, if not exceeded, human levels of accuracy on image classification tasks (31). However, exceedingly large datasets are required to reach these levels of performance - datasets on the order of millions of images in size. For example, Google's in-house dataset, JFT-300M, contains 300 million images scraped from the web with noisy labels (27). These large datasets are being used due to the fact that increasing the size of the dataset generally leads to an increase in model performance. As shown in Figure 1, increasing the number of JFT-300M samples used during training proves a near linear increase with downstream model performance.

Unfortunately a problem arises with obtaining these quantities of data. Training data is usually costly and can be limited in availability. As previously mentioned, the JFT-300M dataset is restricted to only Google engineers, and obtaining a similar sized dataset is confined to massive tech companies who have access to the required funding. Further, these datasets may exist in a limited information domain, such as those consisting of government applications that may require clearances or artist-mimicry where only few works are known to exist.

Similar to image classification models, generative models have also greatly advanced in the last few years. Improving upon Generative Adversarial Networks (GANs), Latent Diffusion Models (LDMs) are the current SOTA for image generation (22). Instead of optimizing over a min-max loss function that is notoriously hard to train, LDMs work by running diffusion in the lower-dimensional latent space - greatly improving the computational efficiency and consequently increasing the accessibility of the model to researchers. LDMs are capable of producing images indistinguishable under the human eye from their actual counterpart. Our work combines the realistic imagery produced by LDMs with the superhuman accuracy of modern image classification frameworks. By using SOTA deep learning generative models, we look to provide a cost efficient way to augment training datasets for improved downstream classification results.

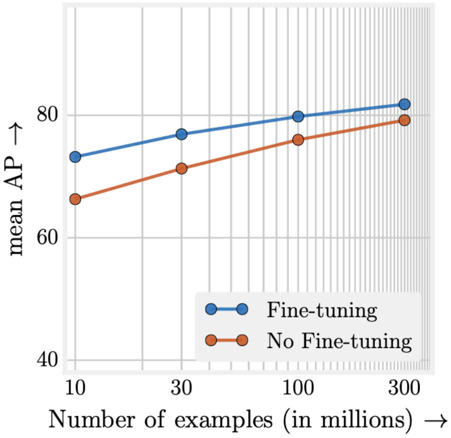

Figure 1. JFT-300M dataset results comparing AP to number of samples available during training. Evaluated on PASCAL dataset (27).

[*] Equal contribution [1] Department of Computer Science, University of Wisconsin-Madison, Madison, Wisconsin [2] Department of Electrical and Computer Engineering, University of Wisconsin-Madison, Madison, Wisconsin. Correspondence to: Derek Paulsen <dpaulsen2@wisc.edu>, Bojun Liu <bliu293@wisc.edu>, Gabriel Gozum <ggozum@wisc.edu>, Joe Gorka <jgorka@wisc.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

1.2. Related Work

Previous works have established the utility of basic dataset augmentation techniques such as warps, flips/rotations, and the injection of (usually Gaussian) noise, for improving performance on image classification, in addition to related tasks such as object detection and image segmentation (25; 28). These basic transformations have the advantage of being very easy to implement with modern machine-learning frameworks such as Tensorflow (1) and PyTorch (18). Drawbacks, however, include increased training time/memory requirements, and the need for manually checking for transformation-induced label changes (25).

Given the ubiquity of standard image transformations, attention has turned to more sophisticated dataset augmentation methods in order to further improve performance. A thorough treatment of these methods can be found in (25). Here we provide a brief summary:

Adversarial training as implemented in works such as (8) and (14) augments images by way of pitting a neural network tasked with classifying images against a second network which attempts to induce classification errors via injection of noise. Though adversarial training is demonstrated in (14) to increase accuracy on adversarial examples, it has not been shown (as far as we are aware) to improve test accuracy.

Neural style transfer as proposed in (6) allows for the creation of synthetic data through combining the ’style’ of one image (day/night, object texture, etc) with the ’content’ of another. While this method has been successfully employed– one interesting example being an object-detection model trained on video-game images, but which generalizes to real-world images (21)–significant effort is often required to choose the relevant ’styles’ and decide to which images they should be applied.

GAN data augmentation, as understood through the original GAN architecture proposed in (7) makes use of a pair of neural networks to generate new images. The ’counterfeiter’ network seeks to generate images that the ’discriminator’ network is unable to distinguish from those belonging to the original dataset. A massive amount of work has been done in this area to improve the architecture and training process (12), but problems remain with training stability and data requirements (24). Despite these drawbacks, however, GAN-generated data has the key ability to automatically generate reasonably realistic data even in highly-specialized domains: for example, in (5) greater than 40% of synthetically-generated images of liver-lesions were identified as ’real’ images by a human expert.

1.3. Contributions

Generative modeling for the purpose of dataset augmentation has yielded impressive results, but remains both an effort and compute-intensive option for improving image-classification performance. This work seeks to provide a middle ground between training one’s own generative model for the purpose of augmenting a dataset, and relying only on basic geometric/photometric transformations of existing images. To do this we harness the power of a large publicly available diffusion-based generative model, taking advantage of the millions of images (and millions of GPU-hours) that went into training it. Specifically, we show that significant accuracy improvements are possible through augmenting an image-classification dataset with synthetic samples generated by the already-trained generative model.

2. Methodology

2.1. Problem Statement

The problem that we address in this paper is image classification in the classic machine learning setting. That is, given training set examples and labels, X^{train}, Y^{train} and a test set of examples and labels, X^{test}, Y^{test}, and classifier f with weights w, we want to find

$$argmin_w \{ \sum_i L(f(w, x_i^{test}), y_i^{test}) \}$$

Where,

$$L(\hat{y}, y) = \begin{cases} 0 \; if \; \hat{y} = y \\ 1 \; otherwise \end{cases}$$

using only the train set to tune w. In plain english, we want to maximize the top-1 accuracy of our model the the test set while only using the train set to perform gradient descent.

2.2. Proposed Solution

Given training set examples and labels, X^{train}, Y^{train}, where each label in the Y^{train} is a natural language string (e.g. $Y^{train} = \{bird, horse, ...\}$) and $x \in X^{train}$ are images our proposed solution works as follows. For each natural language label $y \in Y^{train}$, generate synthetic training examples (x^{syn}, y) by using y as a prompt to a generative image model (e.g. Stable Diffusion). We call these new training examples, X^{syn}, Y^{syn}. We then train a model with X^{train}, Y^{train} and X^{syn}, Y^{syn} as we would normally, applying the same data augmentation, learning rate, and optimizer.

3. Evaluation

In this section we will describe our experiments. We begin with a description of how we generated our synthetic data using a generative model. Next, we describe our procedure for training models. And finally, we present experiments we ran with the generated synthetic data.

3.1. Synthetic Data Generation

In order to generate synthetic data, we leverage Stable Diffusion v1.4 which is publicly available. All images were generated with common default parameters, fp16 precision, resolution of 512 x 512, guidance scale set to 7.5, and iterations set to 50. We attempted to generate images a lower resolution because CIFAR10 images are 32x32, however we found upon visual inspection of the output that there were a significant number of images generated that had no resemblance to the prompt (e.g. 'airplane' would produce seemingly random blobs of color). With the parameters above we generated two synthetic version of CIFAR10 which we call syn1 and syn2. Syn1 was generated with the label names as the prompts (e.g. ”bird”, ”airplane”, etc.), syn2 was generated with a basic prompt template ”a photo of a ¡label name¿”, (e.g. ”a photo of a bird”). Each synthetic dataset took approximately one week of GPU time to generate using a single RTX 2080ti.

3.2. Model Training Procedures

We consider five different models for testing. For each model we tuned the training procedure on the original CIFAR10 dataset, once the tuning was complete we fixed the training procedure and ran experiments, i.e. we made no special tweaks or adjustments for the experiments which use synthetic data train. When tuning the training procedures, we experimented with

- gradient descent algorithms, including relevant parameters for each algorithm
- learning rates and associated parameters
- basic data augmentation, such as random cropping, horizontal flips, etc.

Here we note that the synthetic data had the same basic data augmentation applied to it as the original CIFAR10 data. All experiments were run for 250 epochs.

3.3. Experiments

The first experiment we ran compares the accuracy of training with only CIFAR10 (orig % 100) vs. CIFAR10 with added synthetic data in Figure 2 while varying the model architecture. From these plots we can see that adding in the synthetic data improves the accuracy of the models trained for resnet18, vgg16, and densenet121, although vgg16 shows the smallest performance gain, which may be due to the fact that it was already performing better than the other models. We note that adding both syn1 and syn2 to the training set doesn't improve accuracy and there is no difference between syn1 and syn2.

The next experiment that we ran was to repeat the previous experiment but instead of varying model architectures, we vary the model size (Figure 3). Again we see a similar pattern that all models benefit from adding synthetic training data. Interestingly, we found that resnet50 showed the greatest improvement of all models.

Given the successes of the previous experiments another natural question to ask is if we can achieve similar or better accuracy by swapping of some the original training data with synthetic data. To answer this question we compare the accuracy of models trained with the entire original training dataset vs. using half synthetic half real data. Additionally, we plot using only half of the original data as a reference. For each of these datasets we vary model architectures (Figure 4) and model size (Figure 5). Unfortunately, in all cases swapping half of the original data with the synthetic data significantly degrades model accuracy. Again we see no difference in the performance of syn1 vs. syn2, suggesting that more advance prompt engineering is required to make a significant difference in performance. We note that while swapping half the data for synthetic data didn't achieve the same performance as only using the original dataset, in all cases half synthetic half original data greatly out performed using only half the original data, suggesting that using synthetic data it is possible to reduce the amount of labeled data required to achieve a certain accuracy, although each synthetic example improves accuracy less than each real labeled example.

Finally, we attempted to run only using synthetic data (Figure 6). Here we see that using some amount of real data is required to achieve usable accuracy for a given task, in all cases accuracy was significantly degraded.

The above experimental results have proved the robustness of augmenting diffusion-based synthetic data for the accuracy improvements. To further investigate its significance, we additionally conducted experiments which only employ basic data augmentation techniques, i.e. rotation and vertical flip as benchmarks. We found that if we combined half original data with its rotation-augmented data as well as vertical flip-augmented data, the accuracy cannot be improved anymore (Figure 7). We infer that this is because the original dataset has already employed random horizontal flip augmentation, which reached the limits of the improvements to those traditional augmentation methods that creates new data only based on the existing data. For our diffusion-based augmentation method, it can in advance improve the model going beyond the limits of traditional augmentation strategies (Figure 7).

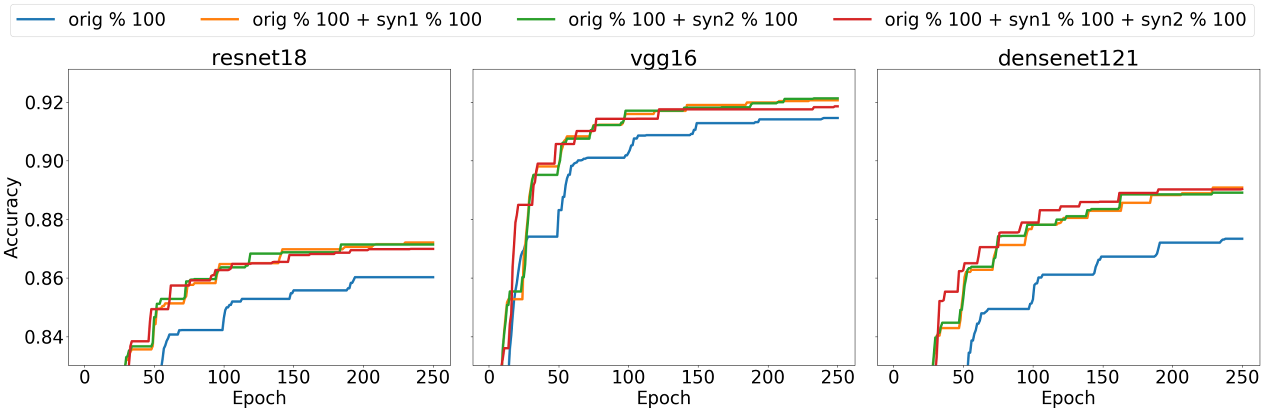

Figure 2. Training with added synthetic data, varying model architectures

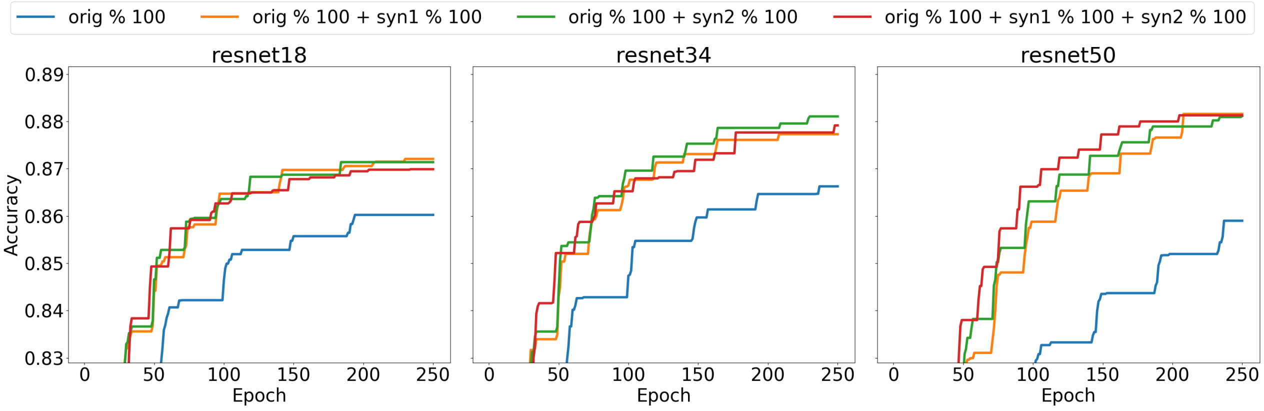

Figure 3. Training with added synthetic data, varying model size

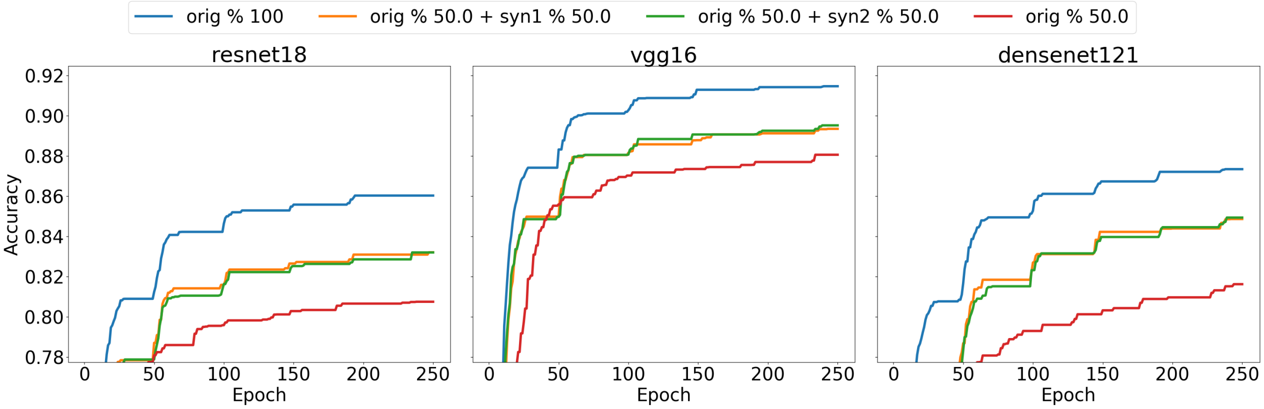

Figure 4. Training with half original data and half synthetic data, varying model architectures

4. Discussion

In this section we discuss the implications of our experimental results.

4.1. Overall Solution

We believe the primary strength of our solution is that it is exceedingly simple to implement, while still giving sig-

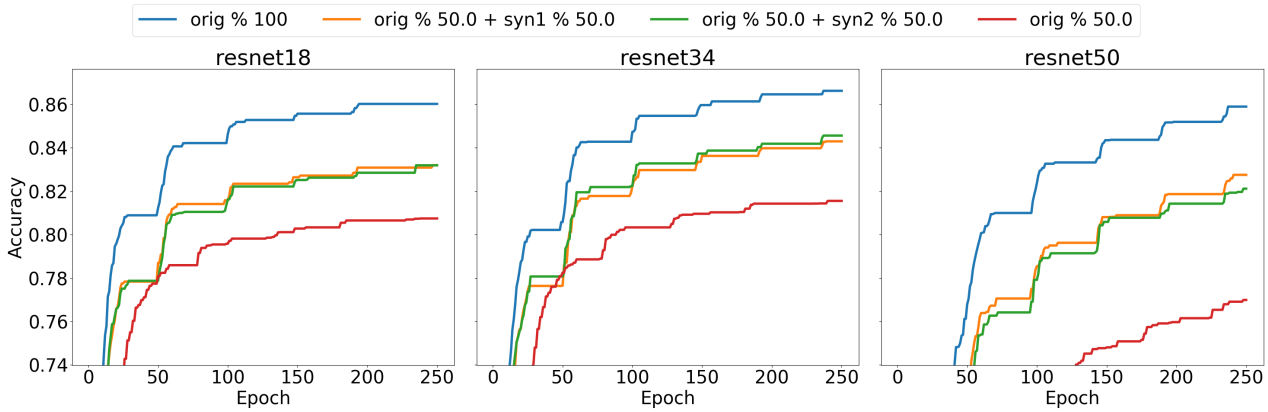

Figure 5. Training with half original data and half synthetic data, varying model size

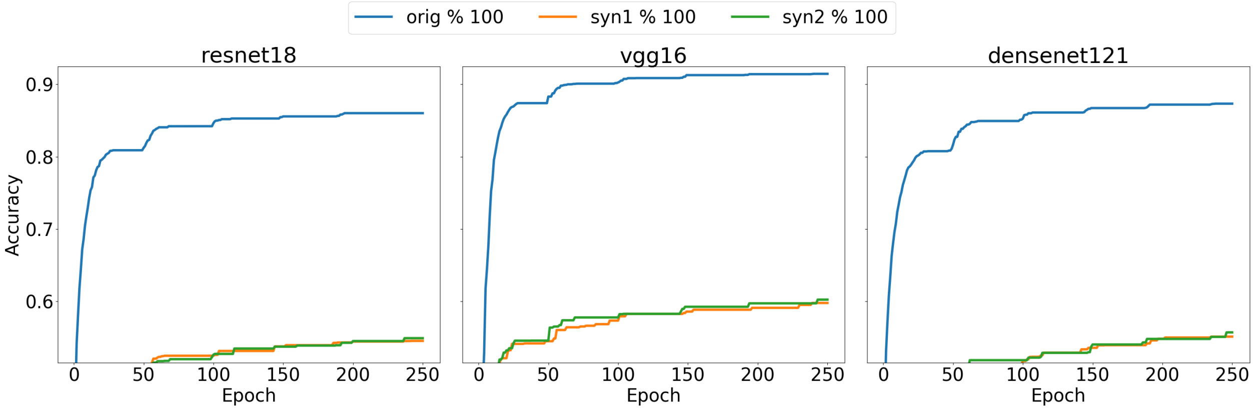

Figure 6. Training with pure synthetic data, varying model architectures

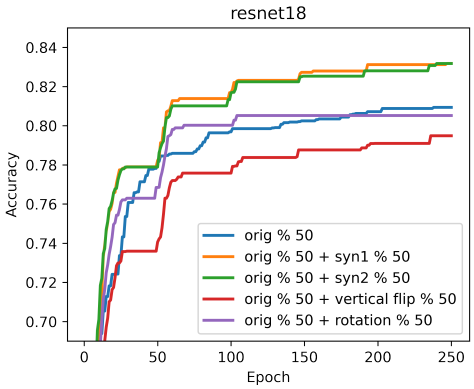

Figure 7. Training with different augmentation strategies based on resnet 18 model

nificant accuracy improvements. In contrast to many other methods which require invasive modifications to the training procedure or model structure, our solution simply requires

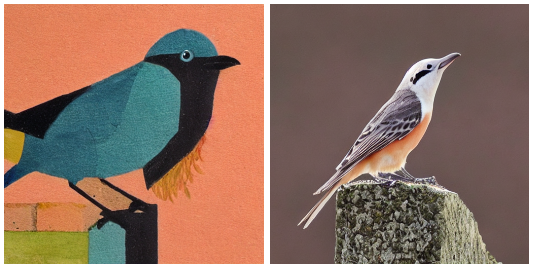

Figure 8. Synthetic images of bird. Left: 'bird'; right: 'A photo of a bird'

data to be generated and then combined with the original training data. As mentioned previously, we generated data with default parameters for stable diffusion, and only attempted minimal prompt engineering. It seems very likely that further tweaking and tuning of the image generation could improve the performance of the solution further, although such tweaks will likely depend on the real labeled data that is used for training in some way.

4.2. Data Generation Time

The most significant weakness to our current solution is the time it takes to generate the synthetic training data. As mentioned previously, each synthetic dataset of 60000 images required approximately a week of GPU time with an RTX 2080ti, which a significant investment in resources. Here we note that this cost is only incurred a single time and it is very easy to estimate up front. In contrast, other data augmentation methods are part of the training procedure, meaning that they incur extra cost every time training is done. Of course, adding extra training data will increase the training time however this increase is relatively modest in comparison to increasing model size or doing other complex data augmentation. Finally, while we had poor results with generating images a lower resolution, it is certainly possible that with additional tuning, the images could generated at a lower resolution which would greatly reduce the computation cost.

4.3. Prompt Engineering

Visual examination of the generated images yielded mostly expected results, that is, most images generated where generic photo-realistic images of the classes. We did notice that in certain cases the model produced output that deviated significantly from this trend when provided only with the label name as the prompt (i.e. syn1). Figure 8 shows two images output for the prompt ”bird”. The image on the right is fairly generic bird however, the image on the left is a much stylized interpretation of the prompt. We found that this variation was removed when switching to the basic prompt template ”a photo of a ¡label name¿”, which was used to generate syn2. We find it interesting that despite this variation, both synthetic datasets seem to perform equivalently in all of our experiments.

5. Future Work

Datasets: In this work, we conducted all experiments based on CIFAR-10 dataset (13). To further validate the applicability of our proposed method, we need to systematically conduct tests on more datasets, such as CIFAR-100 (13), Imagenet (2), etc.

Diffusion-based generative models: The diffusion-based text-to-image generative model we used to generate synthetic data is Stable Diffusion v1.4. We also plan to implement OpenAI’s DALL-E (20), GLIDE (17), DALL-E2 (19) as well as Google Brain’s Imagen (23), etc. These models may produce the datasets with distinct fingerprints that lead to the different performances of data augmentations. We can evaluate these models based on several metrics, e.g. the efficiency of generation process, the proximity of the distributions between synthetic data and real data (e.g. Fréchet inception distance (FID) score (10)), etc and select the one that is most suitable for the diffusion-based dataset augmentation.

Classification models: We evaluated the performance of diffusion-based dataset augmentation on various classification models: resnet 18, resnet 34, resnet 50 (9), vggnet 16 (26) and densenet 121 (11). Interestingly, We found that the largest model - resnet 50 (9) gains the greatest improvement of any model. To further investigate and figure out how the model capacity correlated with the performance-increment by augmentation, we will try larger classification models in the next step. In addition, rather than only applying traditional ConvNet models, we can also employ transformer-based models (29), such as Vision Transformer (ViT) (3) and Swin Transformers (16), which have exhibited remarkable performance in common computer vision tasks and been attracting more and more attention since 2020.

Prompt engineering: The distributions of synthetic datasets of the same subject can be varied by using the prompts with different descriptions, which may as a result affect the performance of augmentation (30). We plan to conduct more tests by designing more prompts with the format ”SUBJECT in the style of STYLE” in which ”SUBJECT” represents the labels of the data while ”STYLE” contains the descriptions you want to add, such as ”photo”, ”art”, etc (15).

Fake images detection: To guarantee that the synthetic data to be augmented are in-distribution and further improve the proximity between the augmented synthetic data and real data, we plan to implement fake images detection techniques in our proposed framework. For example, we can apply out-of-distribution (OOD) detection methods such as Virtual Outlier Synthesis (VOS) (4) to create an outlier that can filter out the synthetic images with bad qualities which are out-of-distribution compared with the original datasets. Also, we can train a binary classifier based on synthetic and real data, and augment those images which are more difficult to be distinguished by the classifier.

Efficiency improvements: The most significant weakness of this work is the time consumed on generating synthetic images. In the future, we can address this issue from two perspectives. Firstly, we can optimize the synthesis procedure by tuning the length of optimization of generations. It is reasonable to assume that the synthetic data that used for data augmentation requires a shorter length of iteration than the synthetic data that required for the human interpretations. And we can also generate the data faster by tuning other hyperparameters in the procedure, e.g. by decreasing the size of the images generated. Secondly, we can improve the efficiency of our method by the effective prevention of the failed augmentations. We need to investigate some strategies that tell whether the synthetic data is going to improve the performance of the model before generating the whole datasets. To enable this, we propose to generate a small sample dataset and employ OOD detection techniques to make an evaluation or test the accuracy of a classification model purely trained on this dataset.

6. Conclusions

In this paper we have explored a novel data augmentation technique using Stable Diffusion. We found that using very little to no prompt engineering was required to create a synthetic dataset which greatly improved accuracy of models trained when compared to only using the original dataset. We think that going forward, this method of data augmentation will not only increase the quality of image classifiers trained but also has the potential to allow classifiers to be utilized by a greater number of deep learning practitioners who don't have resources to acquire hand labeled data. This project has only taken a preliminary look at how SOTA generative models might be used in data augmentation for image classifiers, there is undoubtedly many improvements that can be made to the basic solution template that we have presented in terms of computational efficiency and model accuracy improvement.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[4] Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. Vos: Learning what you don't know by virtual outlier synthesis. *arXiv preprint arXiv:2202.01197*, 2022.

[5] Maayan Frid-Adar, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Gan-based synthetic medical image augmentation for increased CNN performance in liver lesion classification. *CoRR*, abs/1803.01229, 2018.

[6] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style, 2015.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[8] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2014.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] 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. *Advances in neural information processing systems*, 30, 2017.

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[12] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2017.

[13] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[14] Shuangtao Li, Yuanke Chen, Yanlin Peng, and Lin Bai. Learning more robust features with adversarial training, 2018.

[15] Vivian Liu and Lydia B Chilton. Design guidelines for prompt engineering text-to-image generative models. In *CHI Conference on Human Factors in Computing Systems*, pages 1–23, 2022.

[16] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[17] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[18] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[19] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022.

[20] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[21] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games, 2016.

[22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[23] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

[24] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans, 2016.

[25] Connor Shorten and Taghi Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6, 07 2019.

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[27] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era, 2017.

[28] Luke Taylor and Geoff Nitschke. Improving deep learning using generic data augmentation. *CoRR*, abs/1708.06020, 2017.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[30] Sam Witteveen and Martin Andrews. Investigating prompt engineering in diffusion models. *arXiv preprint arXiv:2211.15462*, 2022.

[31] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models, 2022.