Employing Machine Learning Techniques and Frameworks to Aid with Automotive Design

Ryan Santosh*

Oberoi International School, Mumbai, India, 400063

KEYWORDS. Machine Learning, Automotive Design, Stack-Gan, Generative Models.

BRIEF. Modifying existing machine learning techniques and architectures to design a new generative model tailored to automotive design.

ABSTRACT. The use of artificial intelligence and machine learning has seen an exponential rise within the automotive industry, especially within automobile maintenance, supply chain optimisation, and autonomous driving technology; however, machine learning is underexplored within the realm of automotive design, a section of the automotive industry that will benefit significantly from rapid design prototyping enabled by machine learning. We propose a modified stackGAN-based program called AutoForge that uses machine learning techniques to generate novel automotive designs that are high-quality, realistic, and feasible. Our approach enables new design generation at low computational expense with high-quality output.

INTRODUCTION.

The conception of a new automobile model takes around three to five years on average, with newer designs converging upon similar lines [1]. Automobile manufacturers are driven to create a design process that is as swift and efficient as possible while simultaneously improving the quality and creativity of output. However, today's conceptualisation and design process is far too time-consuming, resource-intensive and inefficient [1]. We propose a program that produces high-quality designs by combining image processing techniques to generate designs and natural language processing to increase the output quality by tailoring the design to user-set specifications [2]. Previously, this combination of techniques has proved highly beneficial to creative image generation, with programs such as DALL-E 2 (specialising in image generation from user-made prompt) [3] having used a similar process.

DALL-E 2 is a neural network-based image generation model that creates images from textual descriptions. The model is based on a Transformer architecture, a neural network specialising in processing data sequences, such as text [4]. The process begins with CLIP, Contrastive Language-Image Pre-training, a neural network model that generates image and text encodings for each image-text pair, which captures the meaning of the text. Next, CLIP evaluates the similarity between each image-text pair and iteratively increases the similarity between correct image-text pairs until a suitable output is received [5]. DALL-E 2 then uses a diffusioner before generating image embeddings. Lastly, GLIDE, Guided Language to Image Diffusion for Generation and Editing, takes this information from the generator and produces a low-resolution version of the image. This low-resolution image is then upscaled and run through a series of convolutional layers (upsampling), which increases dimensionality and resolution by increasing pixel count through kernels which filter and refine the image to produce the final product [5].

AutoForge follows a similar approach to a StackGAN Stage- II model [6], trained solely on target designs since it is far less resource intensive, quicker and more feasible. Unlike programs like DALL-E 2, AutoForge will not iteratively update images or use large databases, as seen with the programs above. Nonetheless, the approach used in AutoForge allows for efficient image generation without retraining while consuming relatively limited computational resources [7]. This approach requires minor modifications besides training for the generative image model to provide specific results based on brand identity and distinct design.

To generate high-quality automotive designs, we used modified Stack generative adversarial networks (GANs), initially proposed by Han Zhang et al. [11]. AutoForge uses this approach for high-quality image synthesis due to its efficient and novel training approach, allowing it to produce unique output corresponding to the data it was trained on. GANs were initially proposed in 2014 by Ian Goodfellow et al. [10] and are used within a semi-supervised context. GANs are broken up into two adversarial models, a generative model, which produces plausible data instances and a discriminator model, which learns to effectively distinguish the data produced by the generator from the real training dataset. The generator can only learn through interactions with the discriminator as it cannot access the "real images," while the discriminator can interact with the real and generated images.

To successfully generate new automobile designs, AutoForge must generate high-quality designs that maintain brand design and remain distinct from the training set. Additionally, AutoForge must be versatile and able to produce any form of image based on the text prompt.

METHODOLOGY.

1. Data Set Description. The primary data set used to train AutoForge was entirely custom-made and was gathered from free-use databases published online on platforms such as Kaggle. It includes several images of automobiles, specifically sports cars and their corresponding text pairs. The images are made to be as unique and different in comparison to each other as possible to train the model on multiple designs, shapes, and colours to gain a more versatile and adaptable model. The images in the training database can be altered and made more specific based on the individual styles that a designer or manufacturer may want to be produced. The goal is to generate images similar to the general shape, design and colour of the sports cars in this database but distinct enough to be considered novel. Only images that fit the particular quality and aspect ratio requirements, 1080 x 720 px, were included for training. All images were used for the training of AutoForge. We compiled various types of images into our dataset: concept designs, digital art, photographs, models, and sketches. Various image styles enabled the generative model to produce more diverse designs using a combination of images. Each image from the database is then run through a semantic segmentation algorithm based on DeepLabv3 architecture [8], a convolutional neural network (CNN) architecture used for semantic image segmentation to differentiate between the vehicles and the backgrounds. This particular CNN was chosen since it is the simplest and most efficient model for this task. AutoForge will use pre-trained models exposed to multiple large databases. These models will be further conditioned on automobile designs since this is their primary purpose. While this approach is less efficient regarding zero-shot learning (situations where AutoForge has to adapt to situations it has never been trained for before) than alternative methods; however, it performs well when generating designs.

2. Approach

2.1. The Loss Function and Conditioning. As described previously, GANs contain two adversarial networks; the generator (G) is trained to generate plausible or convincing high-quality images that are difficult for the discriminator to differentiate from the authentic images. On the other hand, discriminator (D) is trained to distinguish between the generated and database images, working directly against the generator's objective.

Loss functions reflect the difference between the images generated by the generator and the real data distribution. For example, equation 1 illustrates a loss function (1), which the generator tries to minimise while the discriminator attempts to maximise:

$$min_G \, max_D V(D,G) = E_{x\sim Pdata}[log\, D(x|y)] + E_{z\sim p_z(z)}[log(1 - D(G(z|y)))] \tag{1}$$

where E_x is the expected value over all real data instances, *x* is a real image from the real data distribution p_{data}, *z* is a noise vector sampled from distribution pz, D(x) is the discriminator's estimate of the probability that x is a real instance, G(z) is the generator's output when given noise *z*, D(G(z)) is the discriminator's estimate of the probability that the fake instance is real, and E_z is the expected value over all random inputs to the generator. Furthermore, since we will provide additional training information, y, to both the discriminator and the generator as an additional input layer, in a process known as conditioning, D(x|y) and G(z|y) are the new inputs. Subsequently, conditioning augmentation techniques are used to generate latent conditioning variables denoted as $\hat{c}$. Conditioning latent variables are input variables incorporated into the generator's network during training to control the generated output. First, the user inputs the text description t, which is encoded through the encoder, producing the text embedding denoted by φt as the input for the generator. Through this technique, latent variables $\hat{c}$ are sampled from the Gaussian distribution:

$$N(\mu(\varphi t), log(\varphi t)) \tag{2}$$

where the mean μ(φt) and the covariance matrix log(φt) are functions of the text embedding φt [12]. From the gaussian distribution **(2),** a random vector z is drawn. Then the generator network transforms the vector by multiplying it with a learned matrix and adding a learned bias vector, giving us a new vector x = G(z). Varying values of z allow GANs to generate diverse outputs with different features and characteristics. Furthermore, the loss function also ensures that the generated images are not too similar to the training batch by using a maximum mean discrepancy loss that compares the similarity between the images produced and those within the database, which the generator attempts to minimise.

2.2. StackGAN Stage-I. As mentioned earlier, the image generation process is split into two parts. In the first stage of the StackGAN approach, AutoForge focuses on producing a low-resolution image, which outlines the basic shape and design of the automobile and colours in the primary colours. In the first stage, AutoForge omits most of the details given in the text prompt since they are not required to make the basic shape of the design.

The first stage of the stackGAN approach trains the discriminator D_1 and the generator G_1 in such a way that it attempts to maximise L_{D_1} and minimise L_{G_1} .

$$L_{D_1} = E_{(I_1,t)\sim Pdata}[\log D_1(I_1, \varphi t)] + E_{z\sim p_z, t\sim Pdata}[log(1 - D_1(G_1(z, \hat{c}_1), \varphi t))] \tag{3}$$

$$L_{G_1} = E_{z\sim p_z, t\sim Pdata}[log(1 - D_1(G_1(z, \hat{c}_1), \varphi t))] + \lambda D_{KL}(N(\mu(\varphi t), \sum(\varphi t))||N(0,1)) \tag{4}$$

Here I_1 is the real image, *t* is the text description from the real data set/distribution p_{data}, and *z* is a random noise vector sampled from the Gaussian distribution. In this model detailed in work by Han Zhang et al. [24], in the generator, the text embedding φt is run through a conditioning augmentation layer which outputs μ_1 which are necessary values for the Gaussian distribution. The conditioning variables are then sampled from the distribution and are concatenated with a N_z dimensional noise vector to generate an image after being run through upsampling blocks. Upsampling and downsampling involve the increase and decrease of dimensionality and size of the input by reducing its spatial resolution.

On the other hand, for the discriminator, the text embedding (generated from the text input) is first compressed and spatially replicated to give us a three-dimensional tensor. At the same time, the image that is generated with the generator is downsampled and analysed, where the discriminator differentiates between the synthetic image and the real image. It then outputs a value, where "1" means the discriminator has identified the synthetic image as a real image and "0" means the discriminator has identified the synthetic image as a synthetic image. If the discriminator cannot identify the synthetic image as fake, the generator has produced good-quality images, while the discriminator must be trained further. The opposite is true if the discriminator could identify the synthetic image as synthetic.

2.3. StackGAN Stage-II. The Stage II GAN works with the results procured from Stage-I, which are used to create higher-quality images. This stage is conditioned on low-resolution images and uses the text prompt inputted by the user to add missing details and correct errors in the image. It uses the information given to add more features and create a complete whole.

The second stage trains the discriminator D_2 and the generator G_2 in such a way that it attempts to maximise L_{D_2} and minimise L_{G_2}.

$$L_{D_2} = E_{(I_2,t)\sim Pdata}[\log D_2(I_2, \varphi t)] + E_{s_1\sim PG_1, t\sim Pdata}[log(1 - D_2(G_2(s_1, \hat{c}_2), \varphi t))] \tag{5}$$

$$L_{G_2} = E_{s_1\sim PG_1, t\sim Pdata}[\log(1 - D_2(G_2(s_1, \hat{c}_2), \varphi t))] + \lambda D_{KL}(N(\mu(\varphi t), \sum(\varphi t))||N(0,1))] \tag{6}$$

The Stage-II GAN follows a different set of formulas compared to the first stage, with slight but significant changes. Firstly, z, the random noise vector, is not used anymore since it is not required, as it is still preserved in the output of stage I. Moreover, the conditioning augmentation layers are also different since the second stage works with low-resolution images as an input and the specific details found in the text prompt. The generator from this stage operates similarly to that of the first one. The first step, relating to text prompt encoding, is kept mainly the same. Differences can be noticed in the second half, where the image result produced by stage-I generator is first run through downsampling blocks. Text and image features are encoded and put through residual blocks, pre-trained to learn multimodal representations. Finally, a high-resolution image is produced after the result is run through upsampling blocks. The discriminator from stage two follows the same structure as that from stage one, except with more downsampling blocks since the image produced during stage two has a larger size and, thus, quality.

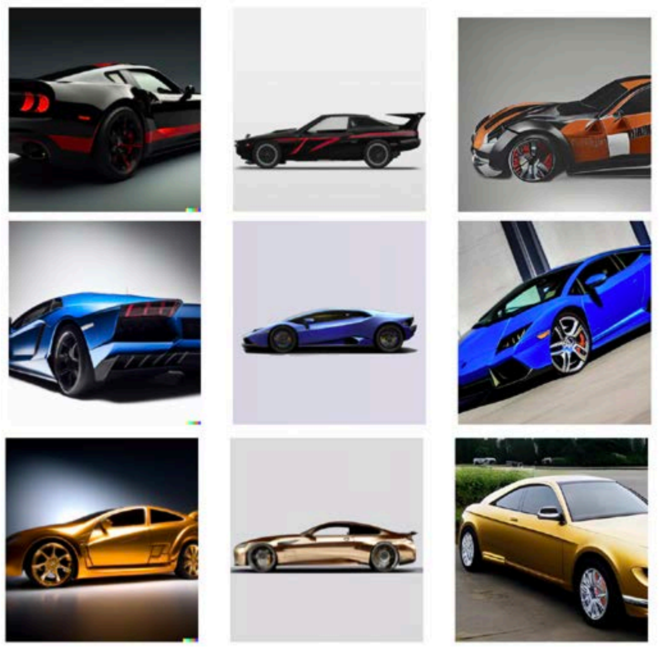

Figure 1. First, three samples were generated by the programs. The first column is full of images generated by DALL-E 2, AutoForge generated the second column, and CLIP-VQGAN generated the third column. The prompts for the first four samples were, "A black sports car with a glossy finish, with a red stripe and spoilers" (Row 1), "A deep blue Lamborghini-style sports car" (Row 2), and "A completely golden coupe with spoilers" (Row 3).

RESULTS AND DISCUSSION.

AutoForge has undergone qualitative evaluations to ensure that this approach can indeed be applied to the automobile industry. We generated eight text prompts to input into these programs to generate the images and then used the samples generated as the baseline for the evaluation and analysis. Comparative tests conducted by human evaluators were used to evaluate the programs and determine the success and quality of their output.

Eight similar types of vehicles from three programs were generated for the primary form of evaluation. The three programs used were DALL-E 2, AutoForge, and the CLIP-VQGAN approach. Our approach is similar to the evaluations detailed in previous papers, where individuals rated each image based on several pre-decided metrics [13]. For the evaluation, generator images were rated by over a hundred participants to determine the applicability of different approaches.

We found that all three methods produce similar images for a given text prompt **(Figure 1)**. For our evaluation, about a hundred human evaluators are selected at random, along with a few professionals from the automotive design industry, to evaluate images produced by these programs based on photorealism, prompt text similarity, consistency, and feasibility. Each image is rated on a scale from zero to one hundred for each given metric. Here zero is the worst a program could perform at a metric, and one hundred is the best. For example: if a user rated an image as seventy per cent for the text prompt similarity metric, the image is considered seventy percent similar to the text prompt. While this method of evaluation may not be objective, it provides a good evaluation of the performance of our program and how it compares with other programs. There are two exceptions in the metrics, however, which will be rated differently. Firstly, for feasibility, experts will rate the designs on how realistic they are. Since a layman cannot make credible evaluations here, we will use professionals from the automobile industry to evaluate the potential designs and their feasibility.

Table 1. A tabular representation of the results from the qualitative evaluations conducted. Refer to supporting information for the other metrics.

Name	Photo-Realism	Prompt Similarity	Overall Impression
DALL-E 2	98.8% ± 3.1%	84.4% ± 7.4%	98.9% ± 2.4%
AutoForge	73.8% ± 22.1%	80.7% ± 15.7%	83.2% ± 8.2%
CLIP+ VQGAN	69.6% ± 18.3%	78.6% ± 21.7%	74.8% ± 7.5%

Table 2. A continuation of the results found from the qualitative tests.

Name	Consistency	Feasibility
Dall-E 2	76.7% ± 6.20%	95.0% ± 9.70%
Our Approach	80.7% ± 15.7%	85.0% ± 12.2%
Clip+VQGAN	78.6% ± 21.7%	76.6% ± 14.7%

Overall, DALL-E 2 generates the most photorealistic, and feasible images with a higher overall impression. In aesthetic quality and photorealism, OpenAI's DALL-E 2 is second to none and has produced stunning automobile designs, which could potentially be used to generate concept images. Based on the evaluations, it has a mean photorealistic value of 98.8% ± 3.10% and a text-prompt similarity of 84.4% ± 7.40% **(Table 1)**. The VQGAN-CLIP approach also produces good-quality automobile images and designs. This approach is worse than the DALL-E 2s program due to the different resources available to both programs. Autoforge performs similarly to CLIP, however holds a slight advantage.

AutoForge had highly successful results; although run with far inferior computational resources (a single quadcore CPU), it has produced relatively high-quality designs. It has been able to compare with other advanced image-generative programs. It is also seen to be more consistent in terms of the shape of the automobile and camera angle **(Table 2)**, allowing it to specialise in the side view design of an automobile, with precise details and a plain background. Autoforge had a mean photorealistic value of 73.8% ± 22.1% and a text-prompt similarity of 80.7% ± 15.7% **(Table 1)**. Finally, based on the overall impression category, wherein evaluators gave their overall perception of the model's design, we find that evaluators liked most designs produced by AutoForge, with an overall impression of 83.2% ± 8.2%. While this is slightly worse than DALL-E 2s results, it performs better than the CLIP model **(Table 1)**.

CONCLUSION AND FUTURE WORKS.

The primary purpose of this research paper was to use machine learning techniques to design a generative model. AutoForge created high-quality automobile designs with low computational costs. This could significantly impact the automobile design process by making it clearer and quicker, producing higher quality and more unique designs. Designers may find it hard to come up with novel designs and express them. Machine learning techniques allow designers to have an open mind, create more diverse and unique automobile designs, and prototype rapidly. The first step toward extending this project would be to introduce more computational resources to engage in further training for the models, including exposing AutoForge to more training images with a wide variety of designs. These changes would make AutoForge far more versatile and capable of producing stunning designs that are realistic and feasible. Further, AutoForge's architecture can be optimised to decrease the time taken to generate images while maintaining or improving the quality of images generated. This is one of the model's weaknesses which ideally should be tackled before moving further.

Finally, AutoForge only produces 2D designs; however, 3D designs could be created to further simplify the design process. The introduction of CAD 3D models in automotive design was revolutionary for the industry and gave designers far more control over their designs, allowing them to create better cars. Now, it is possible to automate this as well, as suggested by the recent research conducted by Saito et al. [14], which has made significant advances in 2D to 3D modelling, as seen in figure 4. The research proposes a multi-level architecture for three-dimensional human digitisation, which can produce highly detailed three-dimensional models of humans. High-quality designs, possibly generated by ML generative models, could be converted into three-dimensional models, which can be used for conceptualisation with some external refining from designers. These approaches will enable streamlined design protocols, wherein design concepts are created by ML programs and refined by professional designers.

REFERENCES

1. C. Bouchard, A. Aoussat, Modelisation of the car design process. *International Journal of Vehicle Design* **31**, 1-5 (2003)
2. K. Crowson, S. Biderman, D. Kornis, D. Stander, E. Hallahan, L. Castricato, E. Raff, Vqgan-clip: Open domain image generation and editing with natural language guidance. arXiv:2204.08583 [Computer Science] (2022)
3. A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, M. Chen, Hierarchical text-conditional image generation with clip latents. arXiv:2204.06125 [Computer Science] (2022)
4. T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, Language Models are Few-Shot Learners. arXiv:2005.14165 [Computer Science] (2020)
5. A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever, Learning transferable visual models from natural language supervision. arXiv:2103.00020 [Computer Science] (2021)
6. H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, D. Metaxes, Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. arXiv:1612.03242 [Computer Science] (2016)
7. R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer, High-resolution image synthesis with latent diffusion models. arXiv:2112.10752 [Computer Science] (2021)
8. L. C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-decoder with atrous separable convolution for semantic image segmentation. arXiv:1802.02611 [Computer Science] (2018)
9. L. Cai, Y. Chen, N. Cai, W. Cheng, H. Wang, Utilizing amarialpha divergence to stabilise the training of generative adversarial networks. *Entropy* **22**, 410 (2020)
10. I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Vengio, Generative adversarial networks. arXiv:1406.2661 [Computer Science] (2014)
11. H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, D. Metaxes, Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. arXiv:1612.03242 [Computer Science] (2016)
12. S. Reed, Z. Akata, X. Yan, L, Logeswaran, B. Schiele, H. Lee, Generative adversarial text to image synthesis. arXiv:1605.05396 [Computer Science] (2016)
13. A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, M. Chen, Hierarchical text-conditional image generation with clip latents. arXiv:2204.06125 [Computer Science] (2022)
14. S. Saito, T. Simon, J. Saragih, H. Joo, PIFuHD: Multi-Level Pixel-Aligned Implicit Function for High-Resolution 3D Human Digitization. arXiv:2004.00452 [Computer Science] (2020)

Ryan Santosh is a student at Oberoi International School in Mumbai, Maharashtra, India.