Unsupervised Learning Methods to Explore Automotive Design

Mason Llewellyn
Stanford University
mason747@stanford.edu

Abstract

Within this paper, we investigate the application of unsupervised learning to image generation, with the intent to create new car designs as a mixture of k existing cars. To accomplish this, we encode images of existing cars into a lower-dimensional latent space, create a new latent vector that represents a convex combination of the existing cars, and finally decode the combined latent vector into a new image. We investigated various methods of compressing vehicles into a lower-dimensional space, including Principal Component Analysis (PCA), a Convolutional Neural Network-based Variational Autoencoder (VAE), and a Resnet18-based VAE. Our methods were evaluated both by their ability to reconstruct a single image of a car, and their ability to design new cars as a mixture of multiple existing cars. The generated images were assessed both on their apparent visual quality and their Frechet Inception Distance to the original dataset of real car images.

1. Introduction

In the century since the automobile's invention, its styling and design have been a core part of its appeal. Automakers use vehicle styling not only to indicate a vehicle's function and improve its recognizability, but also to reference its history. Major auto brands, from high-end marques such as Porsche to affordable ones such as Honda, have developed a cohesive design language that makes them immediately recognizable to the untrained eye. Though unique design elements are used to define a brand's identity, brands often take styling cues from one another. For instance, two of Mazda's most iconic cars, the MX-5 Miata and the RX-7, borrow significantly from other cars' designs. The Miata takes heavy inspiration from contemporary British sports cars while the RX-7 imitates Porsche's 944. A more widespread example would be the fact that LED daytime running lights, which are ubiquitous in nearly all modern cars, were first introduced by Audi in their 2006 R8. Ultimately, stylistic mixing and borrowing has always been a core component of creating new designs.

Figure 1. Example Images from Car Connection Dataset

1.1. Problem Statement

Within our project, we seek to automate the design of new cars as a mixture of existing cars using unsupervised learning methods. We can represent the space of possible automotive designs as a low-dimensional latent space in which every car corresponds to a single point in the space. Using a dataset of automotive images, we intend to learn a mapping from image space to latent space that encodes pertinent design information about the car. Using this latent space we can generate new designs by decoding the latent space representation back into image space. We can generate new designs by sampling random points in the latent space or combine existing designs using convex combinations of encoded latent representations. To measure the realism of our generated car designs, we apply both qualitative evaluations and the Frechet Inception Distance metric as introduced in by Heusel et. al [5].

2. Related Works

2.1. Variational Autoencoder

The main work that our paper relies on is the Variational Autoencoder (VAE) [6]. A variational autoencoder

Figure 2. PCA Reconstruction of an image in the training set

is a neural network architecture that consists of two separate networks: an encoder and a decoder which encode an input into a latent space and decodes it back to an input space. Throughout training, a VAE learns to construct a latent space that encodes meaningful information about the input. Within our work, we apply a variational autoencoder to encode meaningful information about a vehicle's design. Our VAE differs from the original VAE in that it replaces the original VAE's fully-connected encoder and decoder with convolutional neural networks. Our work also experiments with arithmetic in the latent space by using addition to combine the encoded representations of two separate cars into a new design.

2.2. Generative Adversarial Networks

Another line of study that is related to our work is the generation of images using Generative Adversarial Networks (GANs) [3]. GANs pit two neural networks, a generator and a discriminator, against each other in a minimax game. The generator seeks to generate examples imitating the training data and a discriminator seeks to distinguish the real examples from generated ones. Similar to the decoder of a VAE, the generator network of a GAN decodes a vector in latent space into an example in the input space. However, unlike a VAE, the latent vector that is passed into the generator is not encoded from an input image but rather, sampled from a uniform distribution. Though GANs have the ability to learn a mapping from a latent space into the image space, they lack the ability to encode images into that latent space. For this reason, we elect to use VAE's for our design generation over GANs.

2.3. Neural Style Transfer

The task of mixing two separte car designs into one is quite similar to existing work neural style transfer [1]. Neural transfer is the use of a pre-trained neural network to transfer the style of one image onto the content of another. To represent the style of the style image, the authors pass the style image through a pre-trained VGG network and sample the activations in early layers to create a style representation S. The content image is then passed though the same VGG network and later activations are sampled to create the content representation C. To generate the new image, the authors start with a white noise image i and pass it through the VGG network to gain its style and content representations S_i and C_i. The authors then optimize the image using gradient descent to minimize the distance between S_i and S as well as the distance between C_i and C. Though this method yields impressive results it is not exactly what we want for our project. Style transfer gathers low-level style aspects (such as color and texture) from one image which drawing high-level features such as objects from another. In the case of mixing existing cars into a new one, we would like to preseve high and low-level features from both images. For example, we would like our method to both mix low-level features such as vehicle color as well as high-level features such as mixing the wheels of one car with the roofline of another. VAEs do not make this explicit high/low-level distiction between differnt features of an image which is why we use them over style transfer.

3. Data

To train our model, we used the Car Connection Picture Dataset [2], which includes over 64,000 images of vehicles scraped from thecarconnection.com, a vehicle review website. These images come in a variety of sizes but, for our purposes, we resize all of them to 64x64. Though the dataset promises over 60,000 images of cars, visual inspection shows that it is not that simple. Some of the images only included the car's interiors or engines which, for our purposes, would not be very useful. Moreover, many of the images that did include cars, included them within a background which often constituted more image real-estate than the car itself. To ensure that our model focused on the cars' exterior design and not the background or interiors, we elected to filter the dataset of non-car images and crop the images such that they only included cars. Such a large task would be exceedingly tedious to do by hand, so we turned to automated methods. To process our dataset, we used the YOLOv5 (TODO Cite) object detection model to segment the automobile from each individual image and to discard images that were found to not feature cars at all. After this preprocessing, we were left with 45,376 cropped images of cars. More careful processing could have potentially allowed us to discard fewer images of cars, but in the interest of time we elected to simply use the images we had.

4. Methods

4.1. An Initial Baseline: Principal Component Analysis

Definition of Principal Components (1)

$$X_c = X - \mu \tag{2}$$

$$w_{(1)} = \text{argmax}_{\|w\|=1} \|X_c w\|_2 \tag{3}$$

$$= \text{argmax}_{\|w\|=1} (w^T X_c^T X_c w) \tag{4}$$

$$w_{(i)} - i^{\text{th}} \text{ eigenvector of } X_c^T X_c \tag{5}$$

$$S = \begin{bmatrix} | & | & & | \\ w_{(1)} & w_{(2)} & \dots & w_{(k)} \\ | & | & & | \end{bmatrix} \tag{6}$$

The problem of encoding an image into a lower-dimensional latent space can be approached using principal component analysis (PCA). Principal Component Analysis is a method that constructs an orthogonal basis of $k \ll d$ vectors $w_1, \dots w_k$ which spans the given matrix $X \in \mathbb{R}^{d \times n}$ in the directions of greatest variance. By defining images in terms of these k basis vectors, we can convert images from a high-dimensional pixel-based representation to a lower dimensional component-space representation. To convert an image to $X_i \in \mathbb{R}^d$ to a compressed representation $C_i \in \mathbb{R}^k$ we use the following method.

Compression of image using Principal Components (7)

$$C_i = S^T (X_i - \mu) \tag{8}$$

$$X_i = S C_i + \mu \tag{9}$$

To apply this method to a group of images, we first flatten each image to a $3 \times h \times w$ vector and compose a matrix X where each column X_i is a single image. To evaluate the capabilities of PCA, rather than using all 60,000 images from our original dataset, we constrained our training and testing to one model of car: The Porshce 911. The dataset includes images from multiple generations of the 911's production run which all share a similar design language. This means that our compressed representation will have to encode the 911's general design characteristics as well as inter-generational feature changes. We evaluated the PCA model's performace based on its ability to reconstruct images of cars in the dataset. We perfomed PCA on half of the 911 images and evaluated it on the held out set. Our evaluations show that PCA-based reconstruction is very brittle to the original image. Because the principal component vectors span the same subspace as the training examples, any training example can be exactly reconstructed as a linear combination of the principle components. The test images, however, are not linear combinations of the training images and are therefore not able to be represented by the principal component vectors. This brittleness shows that the latent space created by PCA encodes pixel-level information about an image but not high-level design features. This dependence on pixel-level representations also means that even small translations of the original image X_i can drastically change its representation C_i. This means that to create a usable latent space we need a model that is able to encode design-level features and is invariant to image translations.

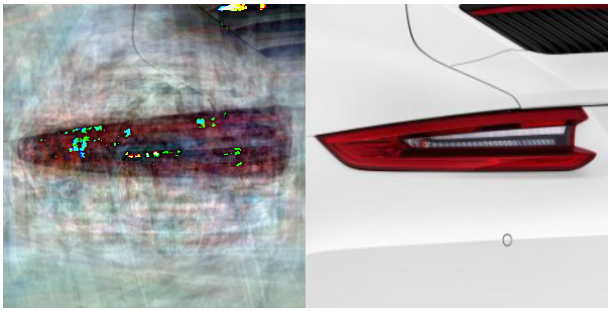

Figure 3. PCA Reconstruction of an unseen image

Layer	Channels	Kernel Size	Stride	Padding	Activation
1	32	3	2	1	LeakyReLU
2	64	3	2	1	LeakyReLU
3	128	3	2	1	LeakyReLU
4	256	3	2	1	LeakyReLU
5	256	3	2	1	LeakyReLU

Table 1. Convolutional Encoder Architecture

Layer	Channels	Kernel Size	Stride	Padding	Activation
1	256	3	2	1	LeakyReLU
2	128	3	2	1	LeakyReLU
3	64	3	2	1	LeakyReLU
4	32	3	2	1	LeakyReLU
5	32	3	2	1	Tanh

Table 2. Convolutional Decoder Architecture

4.1.1 Convolutional Variational Autoencoder

To encode high-level vehicle design features, we turned to a variational autoencoder (VAE) whose encoder and decoder are parameterized by deep convolutional neural networks. Unlike principal component analysis, the multiple layers of convolutional neural networks can encode higher-level image features such as wheels and tires in a way that is not as sensitive to individual pixel values as PCA. The sliding-window method inherent in convolutions also makes convolutions invariant to image translations. The encoder and decoder architectures are described in tables 1, 2.

After being passed through the encoder, the encoded image vector ℓ_i is passed through two learnable linear trans-

layer name	output size	18-layer
conv1	112×112	
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$
	1×1	

Figure 4. Diagram of Resnet-18 Architecture [4]

formations μ and σ to produce the mean and diagonal covariance matrix. The latent representation is then sampled from a normal distribution parameterized by μ and σ. To reconstruct the image, we simply pass the sampled latent vector z_i into our decoder D to produce an output image.

$$\ell_i = E(x_i) \tag{10}$$

$$z_i \sim \mathcal{N}(\mu(\ell_i), \text{Diag}(\sigma(\ell_i))) \tag{11}$$

$$r_i = D(z_i) \tag{12}$$

To train the autoencoder, we apply gradient descent on all parameters to minimize the evidence lower bound (ELBO) loss:

$$\textbf{ELBO}(x_i, z_i) = D_{KL}(q(z|x)\|p(z)) - \mathbb{E}_q[\log p(x|z)] \tag{13}$$

4.1.2 Resenet-18 Variational Autoencoder

In addition to traditional convolutions, we also replace our own convolutional encoder and decoder with an encoder and decoder with the Resnet-18 architecture [4]. The encoder and decoder networks are mirror images of one another. Aside from the change in encoder architecture, the loss and training architecture of the Resnet-18 VAE is the same as the convolutional VAE. Our reasoning for using the Resnet-18 encoder is that its skip connections would ensure that less data is lost as the image is encoded into a latent vector.

5. Experiments

5.1. Training

We trained both of our networks for 30 epochs on our processed dataset of 45,376 images. Our models were trained using and Adam Optimizer with a learning rate of 10^{-4}.

Figure 5. Convolutional VAE Image Reconstruction

Figure 6. Resnet-18 VAE Image Reconstruction

5.2. Evaluation

We evaluate both of our VAEs both based on their ability to reconstruct a given image and create new designs as mixes of existing cars. To mix k existing cars into a new car design we use the encode the k images into k latent vectors and decode the convex sum of the encoded vectors as a single image x_m.

$$\textbf{Mixture of } k \textbf{ Designs} \tag{14}$$

$$z_m = \frac{1}{k}\sum_{i=1}^{k} \mu(E(x_i)) \tag{15}$$

$$x_m = D(z_m) \tag{16}$$

In addition to assessing the quality of reconstructed and mixture images by eye, we use the Frechet Inception Distance to assign a value to the quality of each image. Given a batch of real images X_r and generated images X_g, the Frechet Inception distance provides a quantitative measure of how realistic the generated images are. The FID is calculated by first taking the means μ_r, μ_g and covariances Σ_r and Σ_g of the activations when passing X_r and X_g through an Inception V3 model pretrained on Imagenet. The smaller the FID, the closer the generated images are to the real ones. The final FID is calculated as follows:

$$\textbf{Frechet Inception Distance} \tag{17}$$

$$\text{FID} = \|\mu_r - \mu_g\|_2^2 + \text{trace}\,(\Sigma_r + \Sigma_g - 2\Sigma_r\Sigma_g) \tag{18}$$

6. Results

6.1. Design Reconstruction

Compared to the Resenet VAE, the convolutional VAE produces reconstructions that are more washed out and blurry than the original image. This disparity in quality

VAE	Frechet Inception Distance
Convolutional	80.70
Resnet-18	25.3405

Table 3. Reconstruction Results

Figure 7. Convolutional VAE Combination of two cars

Figure 8. Resnet-18 VAE Succesful combination of two cars

Figure 9. Resnet-18 VAE Unsuccessful combination of two cars

is also reflected in the Convolutional VAE's much higher Frechet Inception Distance (Table 3).

6.2. Design Mixing

6.2.1 2 Cars

As was the case in the reconstruction tasks, the Resnet-18 VAE outperforms the convolutional one in the task of mixing two separate vehicle designs. However, neither of these mixings produced more realistic appearing images than their single-var reconstructions. This is supported by the higher FIDs of two-car mixes in both VAE's compared to their reconstruction FIDs (Table 4) Though the Resnet18 VAE's mixed images are not always realistic, the Convolutional VAE's images rarely (if ever) produce realistic vehicles. The Resnet-18 VAE's successful mixings display properties of both of the vehicles. Figure 8 shows the mix between an Infiniti Q50 sedan and a Mini Cooper Hatchback which results in a vehicle with the body shape of a sedan and the Mini hatchback's blue color. The Resnet-18 VAE's combinations only work best when both vehicles are photographed from the same position. When provided with vehicles in different positions, the Resnet-18 VAE produces warped images that contain features of both original cars but are not recognizable as vehicles.

Figure 10. Resnet18 VAE combination of three cars

Figure 11. Convolutional VAE combination of three cars

VAE	k	Frechet Inception Distance
Convolutional	2	77.96
Convolutional	4	87.31
Resnet-18	2	27.99
Resnet-18	4	32.33

Table 4. k-image mixing results

6.2.2 3 Cars

In the case of both VAE's, combining three cars results in the worst results for both the Convolutional and Resnet-18 VAEs out of all three tests. Aside from the cherry-picked results displayed here (figures 11, 10) attempting to combine three vehicles results in low-quality generated images in almost every case.

7. Conclusion

Our design mixing method works best when all images show cars at the same angle. When the mixed images contain multiple vehicle angles, the result is much more likely to be nonsensical. This phenomenon leads us to question whether single images are a useful representation of vehicle design. Because an image never shows the whole car at once, the model does not learn that there is a relationship between the same car from different angles. This means that, when mixing images of vehicles from different angles, the model struggles to relate the two and produces nonsense.

As we expected, the Resnet-18 VAE significantly outperformed the convolutional VAE both in percieved image quality and Frechet Inception Distance. This is likely due to the increased representational power of the Resenet-based encoders compared to the convolutional one. Future work that builds on this project should investigate if further increases in representational power such as with Resnet-34 or Resnet-50 based encoders would cause similar increases in image quality.

We can potentially address this problem during training

using the labels of the cars as supervision to force cars of the same make, model, and year closer together in latent space and adding the camera angle of the car as part of the encoder's input. This would force the encoder to disentangle the vehicle's design from it's camera angle and allow for higher-quality vehicle mixes. Rather than supervising the latent space, another potential remedy would be to represent vehicles in a way that is not sensitive to camera angle. A collection of images from different camera angles, for example, would show the model the whole vehicle at once, eliminating the entanglement between camera angle and design present in single images. Another angle-agnostic representation would be a 3D point cloud describing the car's shape.

The quality of the dataset could also have impacted the final image quality. As we inspected the 45376 images in our processed dataset, we realized that our set contained duplicate images. This is not a product of our processing method but of there being multiple, duplicate images in the original dataset. The presence of duplicate images means that the number of unique cars in the dataset could be significantly lower than we expected. With fewer unique images in the dataset, the model is not forced to learn a dense latent space that encodes a diverse set of vehicles. This would negatively affect the quality of the mixed images as a sparse latent space would mean that the intermediate representation between k latent vectors is less likely to correspond to a realistic car. Before using this dataset again, we would need to filter our the duplicate images and verify how many unique images we actually have.

Despite the shortcomings of the dataset and the model, we have shown that we can create new car designs from old ones. This lays the groundwork for future works which improve the capabilities of automated car design.

References

[1] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style, 2015. 2

[2] Nicolas Gervais. The car connection dataset. https://github.com/nicolas-gervais/predicting-car-price-from-scraped-data/tree/master/picture-scraper, 219. 2

[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. 2

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 4

[5] 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. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 1

[6] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013. 1

8. Code

The code for both the convolutional and Resnet-18 VAEs was based in-part on code from this blog post.