# Multi-attribute Pizza Generator: Cross-domain Attribute Control with Conditional StyleGAN

Fangda Han[1]
fh199@cs.rutgers.edu

Guoyao Hao[1]
gh343@scarletmail.rutgers.edu

Ricardo Guerrero[2]
r.guerrero@samsung.com

Vladimir Pavlovic[1]
vladimir@cs.rutgers.edu

[1] Rutgers University
Piscataway, NJ, USA

[2] Samsung AI Center
Cambridge, UK

### Abstract

Multi-attribute conditional image generation is a challenging problem in computer vision. We propose **Multi-attribute Pizza Generator (*MPG*)**, a conditional Generative Neural Network (GAN) framework for synthesizing images from a trichotomy of attributes: content, view-geometry, and implicit visual style. We design *MPG* by extending the state-of-the-art StyleGAN2, using a new conditioning technique that guides the intermediate feature maps to learn multi-scale multi-attribute entangled representations of controlling attributes. Because of the complex nature of the multi-attribute image generation problem, we regularize the image generation by predicting the explicit conditioning attributes (ingredients and view). To synthesize a pizza image with view attributes outside the range of natural training images, we design a CGI pizza dataset *PizzaView* using 3D pizza models and employ it to train a view attribute regressor to regularize the generation process, bridging the real and CGI training datasets. To verify the efficacy of *MPG*, we test it on *Pizza10*, a carefully annotated multi-ingredient pizza image dataset. *MPG* can successfully generate photorealistic pizza images with desired ingredients and view attributes, beyond the range of those observed in real-world training data.

## 1 Introduction

A meal appearing in a food image is a product of a complex progression of cooking stages, where multiple ingredients are combined using mechanical and thermo-chemical actions (cooking techniques), followed by physical and stylistic arrangements of food on a plate (plating). Food images are, thus, naturally diverse, displaying complex and highly variable entanglement of shapes, colors, and textures. For instance, pepperoni on a pizza is typically round, but its final appearance depends on the cooking time and cooking temperature. Multiple ingredients mixed together will overlay and obscure each other. A pizza can be rectangular or oval, a pie or a single slice, pictured whole or zoomed in for perceptual or

artistic impact. While Generative Adversarial Networks [5] (GANs) have been used to create high-fidelity images [1, 13, 14] of complex structured objects, such as faces, bodies, birds, cars, etc., they remain challenged by domains such as the food images.

Previous works on food image generation either follow the image-to-image translation paradigm [11, 22] or generate blurry images [6, 31] as a function of a single type of attributes (text of ingredients/instructions). In this work, we propose a new way to extend the capacity of GAN models to solve the problem of generating food images conditioned on the trichotomy of attributes: food content (ingredients), geometric style (view point, scale, horizontal and vertical shift), and visual style (diversity in fine-grained visual appearance of ingredients and the final dish). A main challenge we tackle here is that, for explicit control of different food attributes, no single real food image dataset is sufficiently diverse or fully attribute labeled, to allow learning of photo-realistic conditional attribute models.

To this end, we propose a conditional GAN called Multi-attribute Pizza Generator (*MPG*) inspired by [15]. *MPG* first encodes a set of conditioning attributes, ingredients and geometric view parameters, into entangled scale-specific embeddings, which are then injected into the synthesis network together with the visual "style noise" to create images. To address the lack of simultaneous (paired) labels for food content and geometric style in a single dataset, the *MPG* leverages integration of two new datasets with individual (unpaired) attribute category labels: *Pizza10*, a real-world pizza dataset with ingredient annotations, and *PizzaView*, a CGI dataset created from 3D pizza models annotated with geometric style attributes. *PizzaView* is used to train a geometric style attribute regressor, which subsequently regularizes the conditional generator, trained on *Pizza10*, to generate natural food images with arbitrary view attributes. The implicit visual style is learned in an unsupervised manner.

Our main contributions are: (1) We propose a Multi-attribute conditional GAN framework, which extends StyleGAN2 [15]. The framework incorporates a Multi-scale Multi-attribute Encoder (*MSMAE*) and Attribute Regularizers (*AR*s) that guide generator to synthesize the desired attributes. (2) Starting from pizzaGANdata [22], we create a real-world dataset *Pizza10*, by relabeling a subset of consistent ingredients. *Pizza10* opens up the possibility of learning high-performing multi-ingredient prediction models by eliminating the ubiquitous label noise of [22]. (3) We design a CGI dataset *PizzaView* using 3D pizza models with controlled geometric style attributes (camera angle, scale, x-shift and y-shift), used to train the view attribute regressor[1]. Together, these contributions make it possible to learn photo-realistic food image synthesis models, from disparate (real & CGI) domain attribute data, able to generate images outside the range of natural training image attributes.

The reminder of this manuscript is organized as follows: Sec. 2 introduces the related works including GAN, conditional GAN, food generation and multi-label image generation. Sec. 3 presents our conditional StyleGAN framework *MPG* as well as the Multi-Scale Multi-Attribute Encoder (*MSMAE*) and Attribute Regularizer (*AR*). Sec. 4 and Sec. 5 describe the motivation and methodology to create *Pizza10* and *PizzaView* datasets. Sec. 6 includes experiments that verify the effectiveness of our framework, an ablation study, and an assessment of generated images. Finally, in Sec. 7 we summarize our conclusions.

## 2   Related Works

**GANs** [5] are trained as a two-player min-max game to estimate the true data distribution from samples. In the past few years, GANs have evolved rapidly, improving their ability to

---

[1]Code and *PizzaView* dataset: https://github.com/klory/MPG2.

generate high-fidelity digital images of compact objects and natural scenes [1, 14, 19], as well as human faces [13, 15].

**Conditional GAN** [17] extends GAN by conditioning generation process with auxiliary information to control the appearance of generated images, *e.g.* controlling handwriting digit in MNIST [17], natural object or scene category in ImageNet dataset [18], or the category of CIFAR10 dataset [14]. [21] applies StyleGAN to the logo generation task, our task differs from this as we seek to condition on more than a single label (i.e., discrete ingredients and continuous view attributes); thus, the structures and spatial relationships among labels are far more diverse. GAN-Control [26] also utilizes StyleGAN to control different attributes (*e.g.* human age, pose, illumination, expression and hair color), but they need to split the mapping network into different sub networks for each attribute and construct mini-batch deliberately during training, they also need a second phase to fit individual attribute encoders between latent representations and label per attribute. Our method controls the attributes by encoding the attribute values directly and the model needs to be trained only in a single phase.

Another line of research assumes conditioning in the form of another image (i.e., image-to-image generation such as image editing and/or image inpainting), with the goal of changing its attributes [2, 8] or associated styles [10, 32]. Note the task of our proposed work is complex since different pizzsa often contains different ingredients in terms of shape, color and amount, and we only leverage attributes & labels as the conditioning input, lacking the strong input image prior. Finally, image GANs can be conditioned directly on natural language textual descriptions [16, 29, 30]. While, in principle, more general, these approaches in practice work well on objects such as birds or flowers that share similar structures within classes, but tend to fail on complex scene generation, like those in COCO dataset.

**Food Image Generation** is usually accomplished using a conditional GAN to attain higher fidelity. [11, 22] apply GANs to transform the food type or add/remove ingredients from a source image. This yields food images of reasonable quality, but requires image-based conditioning. In text-based conditioning, [24] concatenate the embedding from a pretrained text encoder with noise to generate a food image. Other approaches, c.f., [4], [6], or [27], employ more sophisticated generation models such as StackGAN2 [30] or ProgressiveGAN [12]. However, these works are only able to generate low-resolution images of food-like texture, but with indistinguishable ingredients and missing details. They also lack comprehensive evaluation metrics, such as the input attribute reconstruction ability in addition to image appearance metrics, like the FID, that provide complementary assessment needed to appraise image realism. Moreover, these methods are unable to control other important image aspects, such as the geometric style. In contrast, our framework is able to synthesize images of high fidelity, with distinct detail in the form of more naturally appearing ingredient elements. We couple this with the ability to explicitly control the view geometry, exceeding the range of view parameters in natural food image datasets. Our extensive evaluation, including using the new conditional FID and mAP scores, affirms this judgement.

**Text-based Image Generation** [30] uses a stacked multi-scale generator to synthesize images based on label embeddings. [29] extends [30] and synthesize details at different subregions of the image by paying attentions to the relevant words in the natural language description. [23] extends [29] by regularizing the generator and redescribing the corresponding text from the synthetic image. While free text is a potentially stronger signal, it is often corrupted by image-unrelated signals, such as writing style. On the other hand, while image labels may appear more specific than free-text, the visual diversity of objects described by those labels, such as food ingredients, and changes in appearance that arise from interactions of those

(a) Generator. **Top-Left**: Multi-Scale Multi-Attribute Encoder (*MSMAE*), each scale has its own layers. Ingredients are represented by a binary vector and then concatenated with view attributes (view angle, scale, x-shift, y-shift). **Bottom-Left**: Mapping Network to encode visual style. **Right**: Synthesis Network. ⊕ means concatenation, notice images at different scales are conditioned with different label embedding

(b) **Left**: Discriminator. **Top right**: Generator Loss, consisting of discriminator loss, ingredients classification loss, and view regressor loss. **Bottom right**: Discriminator Loss, consisting of discriminator losses for real image and fake image.
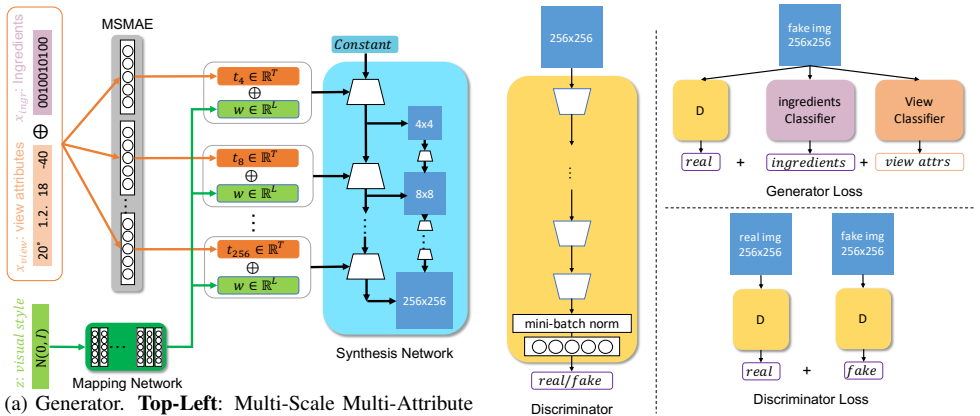
Figure 1: Framework overview

objects can make the multi-label image generation problem an equal if not greater challenge to that of the caption-to-image task.

# 3    Methodology

We adjust StyleGAN2 and create Multi-attribute Pizza Generator (*MPG*), making it possible to generate pizza images with specific, controllable attributes (including ingredients and views) and combinations thereof.

Formally, given a vector **x** indicating the attribute list (Fig. 1(a) Left) and some random visual style **z**, we seek to generate a pizza image **y** that contains the desired attributes. We propose to use a model endowed with Multi-Scale Multi-Attribute Encoder (*MSMAE*) and a Attribute Regularizer (*AR*) to solve this problem.

## 3.1    Multi-Scale Multi-Attribute Encoder

StyleGAN2 transforms and injects style noise **z** into the generator at each scale. Inspired by this, we propose *MSMAE* (Multi-Scale Multi-Attribute Encoder in Fig. 1(a) top-left) to condition the synthesis network on the attributes $\mathbf{x} = \mathbf{x}_{ingr} \oplus \mathbf{x}_{view}$, where $\mathbf{x}_{ingr}$ is the ingredients binary vector, and $\mathbf{x}_{view}$ represents view attributes (i.e., camera angle, scale, x-shift, y-shift).

*MSMAE* consists of a set of sub-encoders $\{Enc_i(\mathbf{x})\}$ (one for each scale $i$) and takes **x** as input to output label embedding $\{\mathbf{t}_i\}$ (also one for each scale $i$). Formally, if the generated image $\tilde{\mathbf{y}}$ is of size $256^2$,

$$\{\mathbf{t}_i\} = \text{MSMAE}(\mathbf{x}) = \{Enc_i(\mathbf{x})\}, i \in \{4, 8, ..., 256\}. \tag{1}$$

We implement $Enc_i$ as a multi-layer perceptron. The mapping network $f$ is kept as in StyleGAN2 to provide diversity to the output image. The label embedding $\{\mathbf{t}_i\}$ is then concatenated with the output from the mapping network together as the input of the synthesis

network $S$. We inject $\mathbf{t}_i \oplus \mathbf{w}$ the same way as in StyleGAN2 and generate the output image $\tilde{\mathbf{y}}$

$$\tilde{\mathbf{y}} = G(\mathbf{x}, \mathbf{z}) = S(\{\mathbf{t}_i \oplus \mathbf{w}\}) = S(\{\text{MSMAE}(\mathbf{x}) \oplus f(\mathbf{z})\}), \tag{2}$$

where G is the generator composed of $\{\text{MSMAE}, f, S\}$.

## 3.2 Attribute Regularizer

To generate pizza image with desired attributes, we use two pretrained model (ingredients classifier $h_{ingr}$ and view attribute regressor $h_{view}$ to regularize training process. An important aspect to note about the *Pizza10* dataset, used to train *MPG*, is that it does not contain view attribute labels, therefore, during training these are imputed with predicted values from the view attribute regressor $h_{view}$, same which is trained exclusively on the CGI *PizzaView* dataset. Refer to Sec. 5 for more details about the performance of $h_{view}$.

**Ingredient Attribute Regularizer.** Inspired by [6, 20], we pretrain a multi-ingredient classifier $h_{ingr}$ on pairs of real image and its ingredient list $\{(\mathbf{y}, \mathbf{x})\}$ with binary cross entropy loss. Then during training the generator, we regularize the generated image $\tilde{\mathbf{y}}$ to predict the correct ingredients $\mathbf{x}_{ingr}$. The Attribute Regularizer for ingredients here can be formalized as

$$\mathcal{L}_{ingr} = \text{BCE}(h_{ingr}(\tilde{\mathbf{y}}), \mathbf{x}_{ingr}). \tag{3}$$

**View Attribute Regularizer.** We apply view attribute regressor $h_{view}$ similarly using L2 loss to regularize the generator as well. Given the view attributes $\mathbf{x}_{view}$ [2] and the synthesized image $\tilde{\mathbf{y}}$, the loss can be formalized as

$$\mathcal{L}_{view} = \text{L2}(h_{view}(\tilde{\mathbf{y}}), \mathbf{x}_{view}). \tag{4}$$

## 3.3 Loss

The generator's objective consists of GAN loss and two attribute regularizations

$$\mathcal{L}_G = -D(G(\mathbf{x}, \mathbf{z})) + \lambda_{ingr}\mathcal{L}_{ingr} + \lambda_{view}\mathcal{L}_{view}, \tag{5}$$

where $\lambda_{ingr}$ and $\lambda_{view}$ are the weights of two attribute regularization terms.

The discriminator's objective consists of GAN loss with R1 regularization [3] applied to penalize the gradient flow from the image to the output, shown to stabilize GAN training, which can be formalized as

$$\mathcal{L}_D = D(G(\mathbf{x}, \mathbf{z})) - D(\mathbf{y}) + \lambda_{r1}\frac{\partial D(\mathbf{x})}{\partial \mathbf{x}}, \tag{6}$$

where $\lambda_{r1}$ is the weight of R1 regularizer.

# 4 Pizza10 Dataset

The performance of neural network models critically depends on the quality of the dataset used to train the models. We begin with pizzaGANdata [22], which contains 9213 pizza

---

[2]Actually, $x_{view}$ is unknown because real images don't contain view labels, we instead use $h_{view}$ trained on *PizzaView* to predict the view attributes as pseudo label $\hat{\mathbf{x}}_{view}$, please read Sec. 5 for more details.
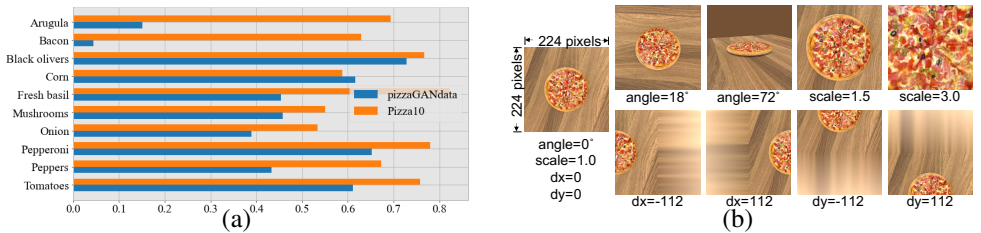
Figure 2: (a) Average precision comparison between pizzaGANdata [22] and *Pizza10* dataset. (b) Example images with different view attributes. Left: Reference image. Right: image transformation along each view attribute

images annotated with 13 ingredients. We manually inspected the labeling in this dataset and found a significant portion of it to be mislabeled. To address this issue, using a custom-created collaborative web platform, we relabeled the dataset and removed three ingredients (*Spinach, Pineapple, Broccoli*) with fewer number of instances (*Corn* is kept since the distinct color and shape makes it a recognizable ingredient even with fewer samples than *Spinach*). The refined dataset, *Pizza10*, decreases the number of 'empty samples' (pizza images with no ingredients labeled) by 40% and increases the number of samples for each ingredient category. We verify the impact of the curated *Pizza10* on the Multi-ingredient Classification task: given a pizza image, we seek to predict its ingredients. We initialized the classifier from ResNet50 [7] with the final fully-connected layer changed to match the output dimension ($\mathbb{R}^{13}$ for pizzaGANdata and $\mathbb{R}^{10}$ for *Pizza10*). The model is trained using binary cross entropy loss, 80% data are used in training while 20% data are for testing. After training, we use the mean average precision (mAP) as the metric.

The results are shown in Fig. 2(a). The same model displays a large performance improvement on the refined *Pizza10* dataset. Upon close inspection, the average precision (AP) for each ingredient increases on most ingredients except *Corn* for the classifier trained on *Pizza10*, therefore demonstrating the importance of careful curating.

# 5  PizzaView Dataset

We also aim to control the view attributes of synthesized images besides ingredients. However, *Pizza10* does not contain view attributes; to train an effective view attribute regressor, we create *PizzaView* from CGI rendered 3D pizza models.

There are two advantages to creating a new pizza dataset from 3D models compared with directly labelling the view attributes in *Pizza10*: (1) The view attributes in *Pizza10* are unevenly distributed, with pizzas usually assuming a center image position, with a small range of scales and viewpoints; these narrow-support attribute distributions will make the training of comprehensive, wide-support view attribute regressors difficult. (2) Manual labeling is laborious and prone to errors, with the labelling process further complicated by ambiguities, such as when the pizza is only partially visible, resulting in high label noise or bias.

To construct *PizzaView*, we first download 40 different 3D pizza models from TurboSquid[3], then import these models into Blender[4] and use Blender's Python API to extract pizza images at different view angles. Without loss of generality, we define the camera viewpoint above the center of the pizza as $0°$, then move the camera between $[0°, 75°]$ on the view hemisphere to generate images with different *view* attributes (we choose $75°$ as the natural

---

[3]https://www.turbosquid.com/
[4]https://www.blender.org/

extreme range of pizzas in *Pizza10*). After rendering the 2D images with various camera angle from 3D models, we further transform the 2D pizza images with *scale*s (range=$[1.0, 3.0]$), x-shift (range=$[-112, 112]$ shown as *dx* in figures) and y-shift (range=$[-112, 112]$, shown as *dy* in figures). We choose this range because the input image size is $224 \times 224$ for the view attribute classifier, so the center of the pizza samples will appear in the image. Fig. 2(b) illustrates the meaning of the four view attributes. These transformations are combined together to create arbitrarily many samples for training of the view attribute regressor.

We use ResNext101 [28] as the backbone, mean square error as the loss term, and batch size of 64, 75% 3D models are used in training while the rest are in validation. The regressor converges after about $150k$ samples. After training, we use root-mean-square error ($RMSE_{attr}$) as the metric and result is shown in Tab. 1 (last row), these errors are very small considering the range of the view attributes.

# 6 Experiments

**Network Structures.** We use the structures in StyleGAN2 and decrease the style noise dimension from 512 to 256. The proposed *MSMAE* is composed of a group of sub-encoders, each sub-encoder is a fully-connected layer with ReLU activation, the label embedding dimension is set to be $T = 256$, hence, after concatenating with style noise $w$, the input to the generator is of the same dimension as that in StyleGAN2.

**Configurations.** For the generator, $\lambda_{ingr} = \lambda_{view} = 1.0$, since there are no view attribute labels in *Pizza10*, we use predicted view attributes as the ground truth label when training the generator. For the discriminator, $\lambda_{r1} = 10$, R1 regularization is performed every 16 minibatches as in StyleGAN2. The learning rate for both generator and discriminator is 0.002.

**Metrics.** We use FID [9] (on 5K images) to assess quality of the generated images; lower FID indicates the generated image distribution is closer to the real image distribution. We use the mean Average Precision [33] (*mAP*) to evaluate the conditioning on the desired ingredients. The ingredient classifier $h_{ingr}$ used is trained on *Pizza10* as shown in Sec. 4 and achieves $mAP = 0.6804$. We finally employ root-mean-square error (RMSE) to evaluate the conditioning on the desired view attributes, where the view regressor $h_{view}$ is trained on *PizzaView*.

## 6.1 Comparison with Baselines

We first compare *MPG* with three works on text-based image generation, where we use the concatenated ingredient names as the text description. View attributes are left out in this comparison. **StackGAN2** [30] extracts the text feature from a pretrained retrieval model then forwards it through a stacked generator with a separate discriminators working at different scales. **CookGAN** [5] also applies StackGAN2 with a cycle-consistent loss to drive the generated images to retrieve its desired ingredients. **AttnGAN** [29] uses a pretrained text encoder to capture the importance of each word and leverages both word-level and sentence-level features to guide the generation process.

Tab. 1 compares between *MPG* and the baselines. *MPG* offers a huge leap forward compared with prior SoTA both in terms of FID and mAP, suggesting it not only generates realistic images with diversity but also improves the conditioning on the desired ingredients.

FID of *MPG* approaches the lower bound of 8.20, which can be estimated from real images[5]. Note that the mAP is much higher on the images generated from *MPG*, even when

---

[5]Obtained by randomly splitting *Pizza10* into two subsets of about 5K images each, then computing the FID

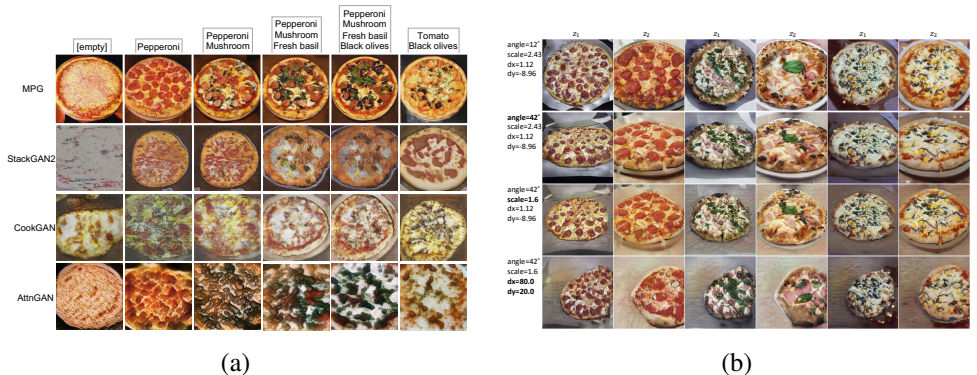(a)                                              (b)

Figure 3: (a) Qualitative comparison between *MPG* and baselines. (b) Images generated using *MPG* from different combinations of ingredients, view attributes and style noise. Images in each row are of the same view attributes and images in each column are of the same style noise. Please read Sec. 6.3 for more details

| Model | FID↓ | mAP$_{ingr}$ ↑ | RMSE$_{angle}$ ↓ | RMSE$_{scale}$ ↓ | RMSE$_{dx}$ ↓ | RMSE$_{dy}$ ↓ |
|---|---|---|---|---|---|---|
| StackGAN2 [50] | 63.51 | 0.3219 | - | - | - | - |
| CookGAN [6] | 45.64 | 0.2796 | - | - | - | - |
| AttnGAN [46] | 74.46 | 0.5729 | - | - | - | - |
| *MPG* | **7.44** | **0.9998** | **3.39** | **0.09** | **4.28** | **4.09** |
| *MPG-MSMAE* | 8.13 | 0.9987 | 3.69 | 0.16 | 4.76 | 4.24 |
| *MPG-MSMAE* * | 9.37 | 0.9867 | 5.32 | 0.21 | 6.94 | 9.12 |
| *MPG-AR* | 7.48 | 0.2404 | 17.04 | 0.34 | 15.54 | 14.94 |
| ingr_cls on *Pizza10* | - | 0.6804 | - | - | - | - |
| view_reg on *PizzaView* | - | - | 1.70 | 0.03 | 1.81 | 1.84 |

Table 1: Quantitative comparison of performances between baselines, *MPG* and its counterparts with missing components

compared with that on real images (0.6804). While this may, at first, appear questionable, we attribute it to the model trained with limited data. There, the synthesized images capture the distinct characteristics of each ingredient but are incapable of covering the real image diversity, making the retrieval easier than that with real images.

Fig. 3(a) shows samples from our *MPG* as well as the baselines conditioned on sample ingredient lists. *MPG* images are subjectively more appealing, with the desired ingredients, compared to the baselines. Notice how *MPG* can effectively add *Pepperoni, Mushroom, Fresh basil and Black olives* one after another into the generated images, more examples are provided in the supplementary materials.

## 6.2 Ablation Study

To verify the design of *MPG*, we consider the following ablation experiments. *MPG-MSMAE* removes Multi-Scale Multi-Attribute Encoder, using the same label embedding for all resolutions. *MPG-MSMAE* * is similar but the label embedding is concatenated with the style noise **z** and forwarded through the mapping network, instead of concatenating it with the output *w* from the mapping network. *MPG-AR* removes the Attribute Regularizer.

From Tab. 1 we observe that removing *MSMAE* increases FID and decreases mAP and RMSE, while removing Attribute Regularizer greatly decreases mAP and RMSE. This is

intuitive, with *AR* the only component to control the attributes during training. In the supplementary materials, we also compare with SeFa [25], which tries to find semantically meaningful dimensions that control attributes in StyleGAN by eigen decomposition.
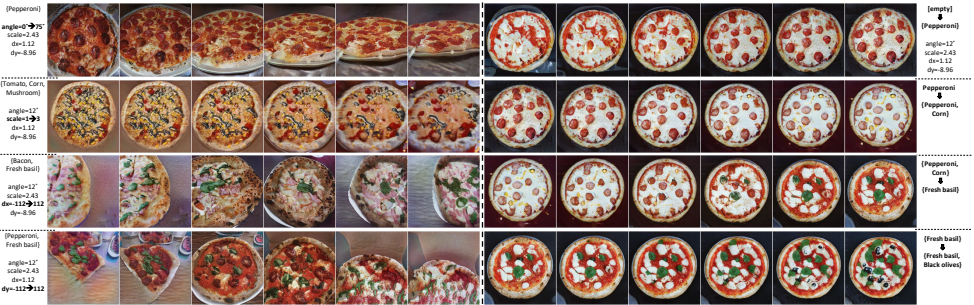


Figure 4: Traversing view attributes (a) and ingredient attributes (b). To traverse the the view attributes (a), images in each row are generated by changing one view attribute at a time (left-most legend column) while fixing the ingredients and the style noise. To traverse the ingredient attributes (b), images in each row are generated by adding or removing select ingredients (right-most column) while keeping the view attributes and the style noise fixed.

## 6.3  Assessment of Generated Images

We provide additional analyses of images created by *MPG*. We first qualitatively study the independence between label embedding and the style noise, then assess the smoothness of the text and the style noise spaces. Additional experiment in the supplementary materials accesses generated image quality with uncommon view attributes more thoroughly.

**Independence.** To analyze the independence between input components, we generate images by fixing each component (i.e., ingredients, view attributes and style noise $z$). The result is shown in Fig. 3(b); images in each row share the same view attributes while each column corresponds to the same style noise $z$. Pairs of adjacent columns are generated with identical ingredients (i.e., {pepperoni}, {bacon, fresh basil}, {mushroom, corn}).

The key observations are: (1) Across each column, view attributes can be modified effectively according to the view attribute input. (2) In each row, images tend to exhibit similar view attributes, differing in color and shape of the pizza and background. These demonstrate that view attribute input and style noise $z$ are independent of each other. (3) By comparing images generated with identical view attributes and style noise $z$ but different ingredients (*e.g.* [row 1, column 1] vs. [row1, column 3] vs [row1, column 5]), we observe that pizzas have similar view attributes, shape and color, the backgrounds are also alike, which demonstrates the independence of the ingredient input and the view attributes and style noise $z$.

**Smoothness.** We verify the smoothness of the learned embedding space by traversing the input space. In Fig. 4 (a), along each row, we fix the ingredients and style noise, while setting the three view attributes to their most frequently predicted values and traverse along one view attribute. It can be seen the targeting view attribute changes smoothly in each row. Note that large $dx$ and $dy$ could result in the change of the image style (*e.g.* pizza shape); we believe this is caused by *Pizza10*, used to train *MPG*, not containing sufficiently many pizza images with similar view attributes. This additionally explains why it is hard to generate images as small as shown in Fig. 2(b) left. In Fig. 4 (b), along each row, we fix the view attributes and style noise and traverse along the ingredient input; the traversing is conducted by treating

the binary indicator existence index as a floating-point number(range=[0.0, 1.0]) and interpolating in between. Each row follows the previous by gradually adding or removing certain ingredients. This suggests the ingredients also change smoothly. Together, this demonstrates the ability of *MPG* to learn, then recreate independent and smooth major factors of variation in this complex image space.

# 7 Conclusion

We propose *MPG*, a Multi-attribute Pizza Generator image synthesis framework, supported by two new datasets *Pizza10* and *PizzaView*, that can effectively learn to create photo-realistic pizza images from combinations of explicit ingredients and geometric view attributes. While effective in visual appearance, the model remains computationally expensive to train, with some loss of detail at fine resolutions, the problems we aim to investigated in future research.

# References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[2] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018.

[3] Harris Drucker and Yann Le Cun. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992.

[4] Ori Bar El, Ori Licht, and Netanel Yosephian. Gilt: Generating images from long text. *arXiv preprint arXiv:1901.02404*, 2019.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[6] Fangda Han, Ricardo Guerrero, and Vladimir Pavlovic. CookGAN: Meal image synthesis from ingredients. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1450–1458, 2020.

[7] 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.

[8] Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. AttGAN: Facial attribute editing by only changing what you want. *IEEE Transactions on Image Processing*, 28(11):5464–5478, 2019.

[9] 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 *Advances in neural information processing systems*, pages 6626–6637, 2017.

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[11] Yoshifumi Ito, Wataru Shimoda, and Keiji Yanai. Food image generation using a large amount of food images with conditional GAN: ramenGAN and recipeGAN. In *Proceedings of the Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management*, pages 71–74, 2018.

[12] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[13] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[14] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33, 2020.

[15] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020.

[16] Bowen Li, Xiaojuan Qi, Thomas Lukasiewicz, and Philip Torr. Controllable text-to-image generation. In *Advances in Neural Information Processing Systems*, pages 2065–2075, 2019.

[17] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[18] Takeru Miyato and Masanori Koyama. cGANs with projection discriminator. *arXiv preprint arXiv:1802.05637*, 2018.

[19] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[20] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *International conference on machine learning*, pages 2642–2651, 2017.

[21] Cedric Oeldorf and Gerasimos Spanakis. LoGANv2: Conditional Style-Based Logo Generation with Generative Adversarial Networks. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 462–468. IEEE, 2019.

[22] Dim P. Papadopoulos, Youssef Tamaazousti, Ferda Ofli, Ingmar Weber, and Antonio Torralba. How to make a pizza: Learning a compositional layer-based GAN model. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[23] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. MirrorGAN: Learning text-to-image generation by redescription. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1505–1514, 2019.

[24] Mark Sabini, Zahra Abdullah, and Darrith Phan. GAN-stronomy: Generative cooking with conditional DCGANs.

[25] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in GANs. *arXiv preprint arXiv:2007.06600*, 2020.

[26] Alon Shoshan, Nadav Bhonker, Igor Kviatkovsky, and Gerard Medioni. GAN-Control: Explicitly Controllable GANs, 2021.

[27] Su Wang, Honghao Gao, Yonghua Zhu, Weilin Zhang, and Yihai Chen. A food dish image generation framework based on progressive growing GANs. In *International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pages 323–333. Springer, 2019.

[28] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[29] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. AttnGAN: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1316–1324, 2018.

[30] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. StackGAN++: Realistic image synthesis with stacked generative adversarial networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1947–1962, 2018.

[31] Bin Zhu and Chong-Wah Ngo. CookGAN: Causality based Text-to-Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5519–5527, 2020.

[32] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[33] Mu Zhu. Recall, precision and average precision. *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, 2:30, 2004.