Adversarial Learning of Expectation and Surprise: Experiments with Geometric Shapes

Oseremen O. Uduehi
School of Electrical Engineering and Computer Science
Ohio University
Athens, OH 45701
ou380517@ohio.edu

Razvan C. Bunescu
Department of Computer Science
University of North Carolina at Charlotte
Charlotte, NC 28223
razvan.bunescu@uncc.edu

Abstract

The ability to produce surprising outputs is a cornerstone of creative behavior. In this paper we propose deep learning architectures that are trained in an adversarial setting to learn patterns of expectations and surprise from data. We introduce benchmark datasets of geometrical shapes that represent well defined patterns of violations of expectations, and use them to verify empirically that the separation between learning of expectations and learning of surprise is essential for achieving good generalization performance.

Introduction and Motivation

Deep generative models based on Generative Adversarial Networks (GAN) (Goodfellow et al. 2014), Variational Aute-Encoders (VAE) (Kingma and Welling 2014; Rezende, Mohamed, and Wierstra 2014), and Transformers (Vaswani et al. 2017) have achieved impressive success in domains ranging from image generation, to music composition, to text generation. The ability of these models to generate highly realistic samples in very diverse modalities is made possible by their capacity to efficiently learn high dimensional distributions from large amounts of training samples. Once trained, the models generate new samples through a random sampling procedure, such as sampling a vector of values from a Gaussian to use as input to the GAN's generator, or sampling the next word to generate according to the distribution computed by a Transformer-based language model. When trained on raw representations of real world objects or artifacts, this *one-model sampling* procedure preserves the structural constraints of the training domain, generating very realistic samples in terms of how they look (e.g. images of faces), read (e.g. prompt-based text generation), or sound (e.g. symbolic music). The generated outputs are also new, in the sense that they do not match any object from the training data, and the model can be used to generate a virtually infinite number of new outputs simply through its random sampling procedure. While generated samples are new, they cannot be said to be original or *surprising*. Being sampled from the learned distribution, they still resemble objects from the training data to a very large extent. As such, while impressive at generating objects that closely resemble, but do not reproduce exactly, the training samples, current approaches to training generative models are severely limited in their ability to generate novel or surprising outputs. Given the importance of novelty and surprise in the design (Yannakakis and Liapis 2016) and evaluation (Grace and Maher 2015) of creative artifacts, this significantly limits the utility of generative models in the area of computational creativity.

To increase the likelihood of surprise, one could eschew the model distribution and sample from a different distribution or entirely at random. However, this results in outputs that are "frequently more frightening than pleasing", due to the lack of control over the "structure / novelty tradeoff" (Todd and Werner 1999). Furthermore, a very high level of surprise can be detrimental to the aim of generating outputs that are aesthetically pleasing. This effect is succinctly captured by Wundt's inverted U-curve, which was later adopted by Berlyne (1971) and others to express the dependency between the hedonic value of a stimulus (e.g. model output) and the novelty of the stimulus as a function that rises to a peak and then falls. Recognizing the necessity of a mechanism to control the generation of surprise while still observing domain-specific structural constraints, Bunescu and Uduehi (2019) introduced a general architecture for *two-model sampling*, wherein an *audience* model is trained to learn patterns of expectations, while a *composer* model learns patterns of surprise, or violations of expectations. By separating the audience learning of expectations from the composer learning of surprise patterns, the proposed Composer-Audience (CA) architecture can generate outputs that confound audience expectations with high probability. This was confirmed in experimental evaluations, where LSTM-based instantiations of the CA architecture were shown to successfully learn patterns of expectation and surprise from distributions of binary sequences.

In this paper, the two-model idea is taken one step closer to computational creativity by using it to learn patterns of expectations and surprise from images of geometric shapes. To this aim, we introduce two GAN-based architectures where the communication of expectations from the audience to the composer is achieved either in the feature space through additive composition, or in the parameter space through norm constraints. Additionally, we establish connections with the domain of style transfer and embed an image-to-image translation module into a third GAN-based CA architecture.

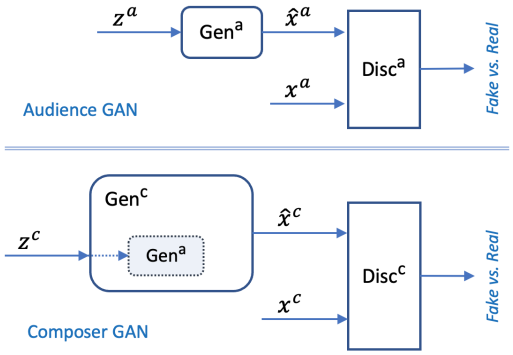

Figure 1: General GAN-based CA architecture.

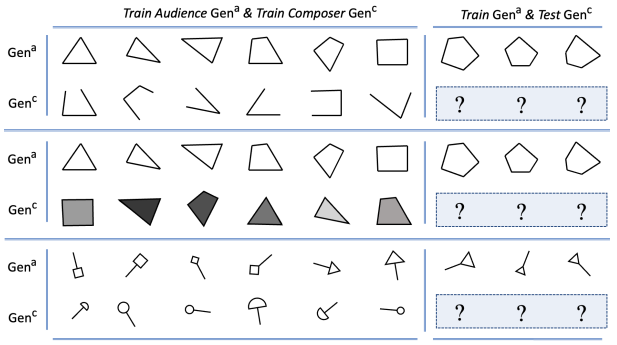

Figure 2: Samples from the 3 datasets used for training the CA models. Top: from complete polygons to polygons with one missing side; Middle: from white polygons to grey polygons; Bottom: from arrows with polygon heads to arrows with full circle or half circle heads.

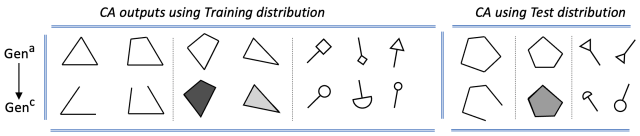

Figure 3: The within-distribution generalization (left) and out-of-distribution generalization (right) expected from a successful CA architecture.

For evaluation purposes, we create benchmark datasets of geometrical figures that capture well defined types of expectations, i.e. empty polygons with 3, 4, and 5 sides, and arrows with various polygons for their heads, and the corresponding surprise, i.e. polygons with one missing side or a grey interior, and arrows with circle or half-circle heads. Experiments demonstrate that the new GAN-based two-model sampling architectures are effective at learning patterns of expectations and surprise in both types of figures. The new GAN-based models are also simpler and more broadly applicable than the style transfer-based instantiations.

Task Definition and Datasets

Overall, the task is to train two generative models: an audience Gen^a that learns a distribution of geometric figures (the expectation), and a composer Gen^c that learns to confound particular expectations in a well-defined, structured way (the surprise). As detailed in the next section and shown in the generic architecture from Figure 1, both the audience and the composer will be trained within the GAN framework as generators that take as input a vector z of random Gaussian values and produce as output an image denoted as $Gen^a(z)$ and $Gen^c(z)$, respectively.

To make the discussion of expectation and surprise more concrete, in Figure 2 we show training samples from each of the three datasets used in the experimental evaluation. The top section shows a CA dataset where the audience is trained to generate polygons with 3, 4, and 5 sides of various relative sizes and orientations. This is the *expectation distribution*, with samples shown in the row labeled Gen^a. The audience model is trained first, and then kept fixed during the training of the composer model. At training time, the composer has access only to a trained audience producing polygons with 3 and 4 sides, and is trained to generate the same kind of polygons, but with a missing side. This is the *surprise distribution*, with samples shown in the row labeled Gen^c. The training is unpaired, e.g. the composer is not given the complete triangle corresponding to the incomplete triangle that it is trained to generate; given that the vertices of the polygons in the training data were generated at random, the corresponding complete triangle is very likely to not even be present in the dataset on which the audience was trained. At test time, the composer is given access to an audience model producing a new type of expectation distribution: polygons with 5 sides. Even though the composer has not seen any pentagons during training, the generalization task is considered successful if the composer can produce the same kind of surprise for pentagons too, i.e. generate the same pentagon as the audience, but with one missing side, as shown in the right section of Figure 3.

The middle section of Figure 2 shows samples from a second dataset where the audience distribution is the same as in the first dataset, i.e. polygons with 3, 4, and 5 sides. However, the composer is tasked with learning to confound the expectation of a white interior by training solely on triangles and quadrilaterals that are colored with random shades of gray. As in the first experiment, the composer is deemed successful if it generalizes to pentagons, a type of polygons that it has never seen at training time, i.e. when used with an audience model that was trained to generate white pentagons, the composer should apply the same surprise pattern and generate gray pentagons.

The bottom section of Figure 2 shows samples from the third dataset, where the audience is trained on a distribution of arrows with 4 types of heads: square, diamond, triangle, and inverted triangle. The arrows can appear at any orientation, with a tail always the same length, whereas the size of the head can vary. The composer confounds the head expectations by replacing the polygon shaped head generated by the audience with either a full circle or half circle head, at random. At training time, the composer sees only square, diamond, and triangle heads, as they are generated by the audience model. At test time, the composer needs to

thwart the audience expectations by replacing the previously unseen inverted triangle heads with full or half circles.

In Figure 3 we show the two kinds of generalization performance that are expected from a CA architecture to be deemed successful. Taking the first dataset as an example, given a random Gaussian vector z, the audience model generates a triangle or a quadrilateral and the composer generates the *same* polygon, but with a missing side, as shown in the first two columns in the figure. Thus, the composer preserves the polygon expectation from the audience, but also surprises the audience by missing one side. We call this *within-distribution* generalization, because the composer generates surprise for types of polygons (3 and 4 sides) that it has seen at training time. In contrast, the right-hand side of Figure 3 shows examples of *out-of-distribution* (OOD) generalization. For the first dataset, this requires the composer to generate surprise by missing an edge from pentagons generated (expected) by the audience, in the context where pentagons were never used to train the composer.

Connections to Other Tasks The CA's generation of surprising features while preserving overall expectations, as illustrated by the within-distribution generalization examples from Figure 3, bears similarities with the mappings performed by unsupervised image-to-image translation and style transfer models, such as CoGAN (Liu and Tuzel 2016), CycleGAN (Zhu et al. 2017), Augmented CycleGAN (Almahairi et al. 2018), or MUNIT (Huang et al. 2018), to name just a few. There are however fundamental differences between learning to surprise in CA models and image-to-image translation. A CA model generates an output image from scratch, e.g. a Gaussian sample, whereas style transfer models use an existing image as input. The CA model's ultimate objective is to generate truly novel outputs by confounding the expectations of an audience not seen by the composer during training, as shown in the OOD generalization examples from Figure 3, whereas image-to-image translation is traditionally aimed for within-distribution generalization. CycleGAN, for example, documents its limitations on OOD samples and geometric transformations. The requirement that CA architectures learn patterns of surprise that generalize to OOD audiences means they can benefit from domain shift invariance, e.g. Adversarial Discriminative Domain Adaptation (Tzeng et al. 2017), or OOD invariance, e.g. Invariant Risk Minimization (Arjovsky et al. 2020) (but see Rosenfeld et al. (2021) for limitations).

GAN-based Architectures for Surprise

Figure 1 shows the proposed generic GAN-based approach for the CA architecture. The audience model, shown at the top, is instantiated as a prototypical GAN: a standard Gaussian vector z^a is used as input to a Generator network Gen^a that outputs an image $\hat{x}^a$, whereas a Discriminator network $Disc^a$ takes real images x^a and generated images $\hat{x}^a$ as input and is trained to determine whether they are fake or real. The composer GAN, shown at the bottom, has a similar architecture, with one important difference: its Generator network Gen^c uses the *expectations* computed by the audience generator Gen^a. Analogous to the kind of expectations that

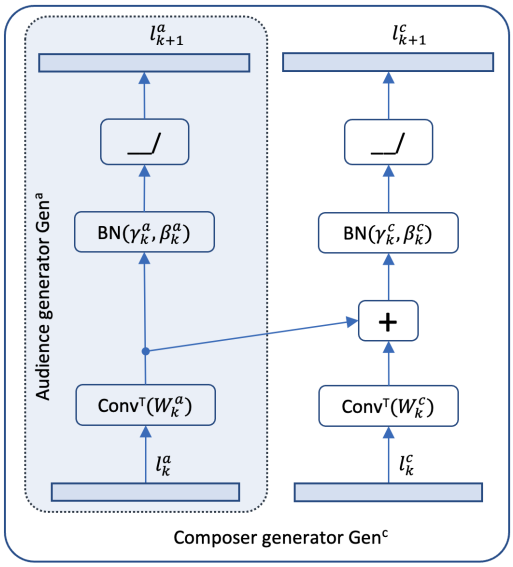

Figure 4: Additive layer-based communication of expectations from the audience (left, shaded) to the composer.

are engineered by composers in their music , here we use the term *expectation* in its broader sense to refer to values that are more likely to appear than other values, according to an audience model. These values can refer to the audience model outputs, layers, or even parameters.

There can be many implementations of this generic architecture, depending on how expectations are communicated between the audience and the composer generators. Here, we introduce 3 instantiations:

1. **Layer** based expectations: The composer uses all the layers computed by the audience generator.
2. **Parameter** based expectations: The composer is constrained to be close to the audience parameters.
3. **Output** based expectations: The composer uses the output computed by the audience generator.

In both the layer-based and parameter-based instantiations, the composer generator is set to have the same architecture as the audience generator, i.e. a sequence of transposed convolution layers. The layer-based version is shown in Figure 4, where we chose to communicate expectations by adding at every layer l_k the audience output of the convolution operation to the corresponding convolution output from the composer. The resulting sum is then passed through the usual batch normalization operation, followed by the application of a nonlinear activation function. Note that the addition can theoretically be implemented at any of 3 distinct places, each with a potentially different behavior: before convolution, before normalization, or before activation. The summation is used only in the composer, whereas the audience processes the input as if it were run separately, on its own. Given an input latent vector z^c, this means that the composer has access to what the audience would have produced for that input, at every layer. An alternative is to use the summation also as input to the normalization operator in

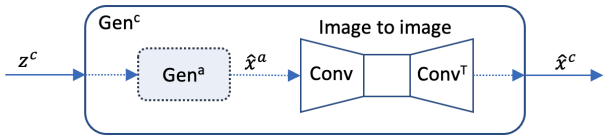

Figure 5: Output-based communication of expectations from the audience generator (left, shaded) to the composer.

the audience generator, however this was less effective during evaluations, likely due to the audience parameters being used on distributions unseen during training.

In the parameter-based version, we consider that the audience expectations are captured through its parameters W_k^a at every layer k. While the composer's job is to learn patterns of surprise from data, it is still required to preserve many of the expectations produced by the audience, e.g. it still needs to produce the same kind of shapes as the audience model. We implement this by initializing the composer generator as $W_k^c = W_k^a$ and then fine-tuning the composer while requiring that its parameters do not diverge too much from the audience. This is done by imposing a constraint on the L_2 norm of the difference between parameters, i.e. by adding the term $||W_k^c - W_k^a||^2$ to the GAN objective function.

In the output-based version shown in Figure 5, the composer accesses the audience expectations only as they are produced in its final output image. This is implemented by passing the audience output to an image-to-image translation network that is trained to output the composer distribution.

Connections to Related Work In the LSTM-based CA architecture of Bunescu and Uduehi (2019), audience expectations are communicated to the composer LSTM using the output probability distribution of the audience LSTM at each time step. As such, it can be seen as using an Output-based style of communicating expectations. Todd and Werner (1999) survey evolutionary approaches to music composition where a *composer*'s fitness function is guided by judgements that are elicited from a *critic* every time a new generation of composer models or composer outputs is to be generated. The critic can be a human (unfeasible), rule-based (brittle, fixed aesthetic criteria), learned (from a human critic decisions), or co-evolved with the composer to prefer songs that violate its current expectations. The surprise preference is encoded directly in the fitness function as a difference between the probabilities of the expected and observed notes. With the sole exception of the critic initialization of expectations, which are calculated from a collection of simple folk tune melodies, this evolutionary *composer-critic* approach generates "musical sequences" entirely from scratch, evolving its own, largely unconstrained aesthetics that "the human user would find worthless". In contrast, our *composer-audience* approach to generation of surprise is entirely data-driven: structural constraints and expectations are learned from the audience dataset, whereas patterns of surprise reflect the hedonic values implicit in the composer dataset. As such, the CA model learns the aesthetics manifest in the data, be it human or machine generated. Todd and Werner (1999) mention the potential utility of modeling expectation-violation with respect to not only the exposure to previous songs, but also relative to the expectations engendered within the current song. This parallels the long- vs. short-term distinction proposed by Pearce, Conklin, and Wiggins (2004) for the statistical prediction of monophonic music. While modern techniques such as self-attention (Vaswani et al. 2017) may obviate the need for explicit short-term models for prediction, we believe within-output expectations are still important for the task of generating surprise.

Experiments and Discussion

We implement[1] the image-to-image component of the Output-based approach using the Augmented CycleGAN. For the audience and composer generators in the Layer-based and Param-based approaches, we use an architecture that starts with 3 transposed convolution layers, followed by 6 ResNet blocks, and ending with 2 more transposed convolution layers. Since a CycleGAN requires training of two generators, the Output-based models require double the number of parameters when compared with the Layer and Param based models. LSGAN loss is used during training. To make the audience identifiable between training and testing of the composer, we use a one-hot vector as input to encode the type of shape (e.g. 3, 4, or 5 sides).

As described in the previous section, an audience model is trained first, and then kept fixed while training the composer model, so that the audience expectations do not change. Samples of this separate training of the audience and composer models are shown in Figure 6 for each of the three datasets. On the missing side surprise task, the Layer-based composer obtains the best OOD generalization, as it is able to successfully remove a random edge from pentagons, a type of polygon that it has not seen during training. The Output and Param based composers achieve this OOD generalization with various degrees of success, sometimes removing more than one edge. On the gray interior surprise task, the best OOD generalization is obtained by the Output-based composer, whereas the Layer and Param based approaches have difficulties in preserving the structural expectations, i.e. the actual polygon shape, generated by the corresponding audience model. Finally, on the arrowhead surprise task, all three types of composer models appear to work well, with the Layer-based model slightly edging the other two models in terms of the quality of circle and half-circle heads it generates for both within and out of distribution evaluations.

To determine the importance of keeping the audience model fixed during composer training, in Figure 7 we show samples from experiments where the audience expectations are allowed to change when training the composer. For the Layer-based approach, this means that the audience parameters change when the GAN loss is backpropagated through both the audience generator Gen^a and the composer generator Gen^c. The analogue for the Param-based approach was to remove the L2 regularization term $||W_k^c - W_k^a||^2$ from the GAN objective, which means that the composer parameters are not longer required to be close to the audience parameters. The samples from Figure 7 show that in this

[1] https://github.com/uoseremen/ComposerAudienceGAN