Understanding Innovation Engines: Automated Creativity and Improved Stochastic Optimization via Deep Learning

A. Nguyen anguyen8@uwyo.edu
University of Wyoming

J. Yosinski jason@geometricintelligence.com
Cornell University & Geometric Intelligence

J. Clune jeffclune@uwyo.edu
University of Wyoming

Abstract

The Achilles Heel of stochastic optimization algorithms is getting trapped on local optima. Novelty Search mitigates this problem by encouraging exploration in all interesting directions by replacing the performance objective with a reward for novel behaviors. This reward for novel behaviors has traditionally required a human-crafted, behavioral distance function. While Novelty Search is a major conceptual breakthrough and outperforms traditional stochastic optimization on certain problems, it is not clear how to apply it to challenging, high-dimensional problems where specifying a useful behavioral distance function is difficult. For example, in the space of images, how do you encourage novelty to produce hawks and heroes instead of endless pixel static? Here we propose a new algorithm, the Innovation Engine, that builds on Novelty Search by replacing the human-crafted behavioral distance with a Deep Neural Network (DNN) that can recognize *interesting* differences between phenotypes. The key insight is that DNNs can recognize similarities and differences between phenotypes at an abstract level, wherein novelty means *interesting* novelty. For example, a DNN-based novelty search in the image space does not explore in the low-level pixel space, but instead creates a pressure to create new *types* of images (e.g. churches, mosques, obelisks, etc.). Here we describe the long-term vision for the Innovation Engine algorithm, which involves many technical challenges that remain to be solved. We then implement a simplified version of the algorithm that enables us to explore some of the algorithm's key motivations. Our initial results, in the domain of images, suggest that Innovation Engines could ultimately automate the production of endless streams of interesting solutions in any domain: e.g. producing intelligent software, robot controllers, optimized physical components, and art.

Keywords

Genetic algorithms, deep neural networks, CPPNs, MAP-Elites.

1 Introduction[1]

Stochastic optimization and search algorithms, such as simulated annealing and evolutionary algorithms (EAs), often outperform human engineers in several domains (Koza

[1]This paper is an extended version of a conference paper (Nguyen et al., 2015b) that includes many new analyses and experiments described in Section 5.

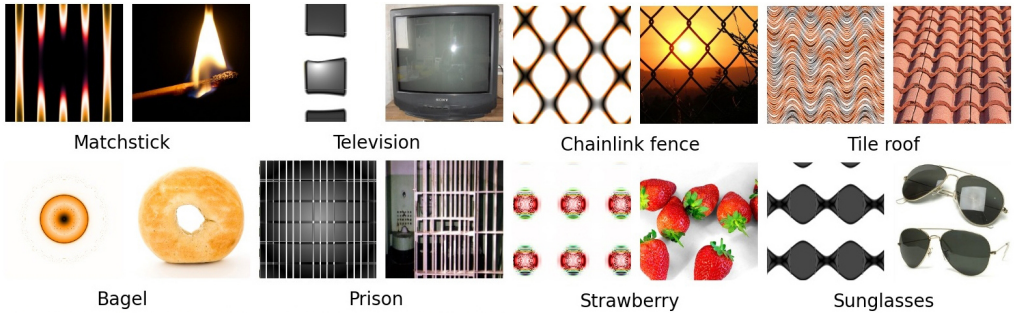


Figure 1: Images produced by an Innovation Engine that look like example target classes. In each pair, an evolved image (left) is shown with a real image (right) from the training set used to train the deep neural network that evaluates evolving images.

et al., 2005). However, there are other domains in which these algorithms cannot produce effective solutions yet. Their Achilles Heel is the trap of local optima (Woolley and Stanley, 2011), where the objective given to an algorithm (e.g. a fitness function) prevents the search from leaving sub-optimal solutions and reaching better ones. Novelty Search (Lehman and Stanley, 2008, 2011a) addresses this problem by collecting the stepping stones needed to ultimately lead to an objective instead of directly optimizing towards it. The algorithm encourages searching in all directions by replacing a performance objective with a reward for novel behaviors, the novelty of which is measured with a distance function in the behavior space (Li et al., 2014). This recent conceptual breakthrough has been shown to outperform traditional stochastic optimization on deceptive problems where specifying distances between desired behaviors is easy (Lehman and Stanley, 2008, 2011a). Reducing a high-dimensional search space to a low-dimensional one is essential to the success of Novelty Search, because in high-dimensional search spaces there are too many ways to be novel without being interesting (Cuccu and Gomez, 2011). For example, if novelty is measured directly in the high-dimensional space of pixels in a 60,000 pixel image, being different can mean different static patterns, which are not *interestingly different types* of images.

Here we propose a novel algorithm called an *Innovation Engine* that enables searching in high-dimensional spaces for which it is difficult for humans to define what constitutes *interestingly different behaviors*. The key insight is to use a deep neural network (DNN) (Bengio, 2009) as the evaluation function to reduce a high-dimensional search space to a low-dimensional search space where novelty means *interesting* novelty. State-of-the-art DNNs have demonstrated impressive and sometimes human-competitive results on many pattern recognition tasks (Krizhevsky et al., 2012; Bengio, 2009). They see past the myriad pixel differences, such as lighting changes, rotations, zooms, and occlusions, to recognize abstract concepts in images, such as tigers, tables, and turnips. Here we suggest harnessing the power of DNNs to recognize different types of things in the abstract, high-level spaces they can make distinctions in. A second reason for choosing DNNs is that they work by hierarchically recognizing features. In images, for example, they recognize faces by combining edges into corners, then corners into eyes or noses, and then they combine these features into even higher-level features such as faces (Nguyen et al., 2016b; Yosinski et al., 2015; Zeiler and Fergus, 2014; Nguyen et al., 2016a). Such a hierarchy of features is beneficial because those features can be produced in different combinations to produce new types of ideas/solutions.

Despite their impressive performance, DNNs can also make mistakes. Szegedy et al. (2014) found that it is possible to add imperceptible changes to an image originally correctly classified (e.g. as a bell pepper) such that a DNN will label it as something else entirely (e.g. an ostrich). Nguyen et al. (2015a) showed a different, but related, problem: images can be synthesized from scratch that are completely unrecognizable to human eyes as familiar objects, but that DNNs label with near-certainty as common objects (e.g. DNNs will declare with certainty that a picture filled with white noise static is an armadillo). While such shortcomings of DNNs impair Innovation Engines a fraction of the time, in this paper we emphasize that remaining fraction of the time wherein using DNNs as evaluators works well. Innovation Engines will only improve as DNNs are redesigned to not be so easily fooled.

We first describe our long-term, ultimate vision for Innovation Engines that require no labeled data to endlessly innovate in any domain. Because there are many technical hurdles to overcome to reach that vision, we also describe a simpler, version 1.0 Innovation Engine that harnesses labeled data to simulate how the ultimate Innovation Engine might function. While Innovation Engines should work in any domain, we test one in the image generating domain that originally inspired the Novelty Search algorithm (Stanley and Lehman, 2015) and show that it can automatically produce a diversity of interesting images (Fig. 1). We also confirm some expectations regarding why Innovation Engines are expected to work.

2 Innovation Engines

The Innovation Engine algorithm seeks to abstract the process of curiosity and habituation that occurs in humans. Historically, humans create ideas based on combinations of, or changes to, previous ideas, evaluate whether these ideas are interesting, and retain the interesting ideas to create more advanced ideas (Fig. 2). We propose to automate the entire process by having stochastic optimization (e.g. an evolutionary algorithm) generate new behaviors and a DNN evaluate whether the behaviors are interestingly new. The DNN will then be re-trained to learn all behaviors generated so far and evolution will be asked to produce new behaviors that the network has not seen before. This algorithm should be able to automatically create an endless stream of interesting solutions in any domain, e.g. producing robot controllers, optimized electrical circuits, and even art.

Creating an Innovation Engine requires generating and retaining "stepping stones to everywhere." The stepping stones on the path to any particular innovation are not known ahead of time (Lehman and Stanley, 2011a). From the stone age, for example, the path to create a telephone did not involve inventing only things that improved long-distance communication, but instead involved accumulating all interesting innovations (Fig. 2). In fact, had human culture been restricted to only producing inventions that improve long-distance communication, it is likely that the telephone would never have been developed. That is because many of the fundamental telephone-enabling inventions were not invented because they enabled long-distance communication (e.g. wires, electricity, electromagnets, etc.), but instead were invented at the time for other purposes. The same is true for nearly every significant invention in human history: many of the key enabling technologies were originally invented for other purposes (Lehman and Stanley, 2011b). In art, just as in science, there is a similar accumulation of interesting ideas over time and a pressure to "make something new", which leads to a steady discovery of new artistic ideas over time (Lehman and Stanley, 2011b). Human culture, therefore, can be seen as an "Innovation Engine" that steadily produces new

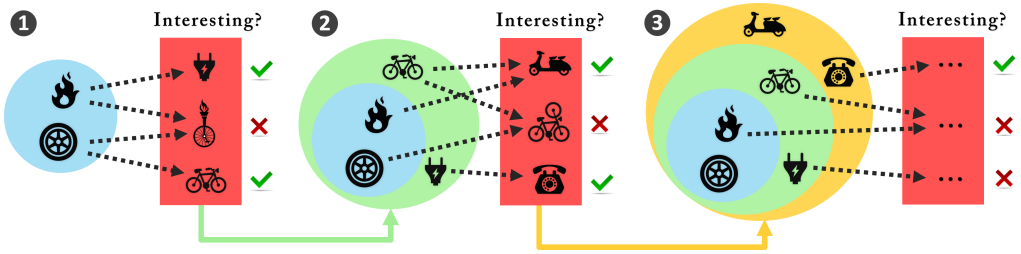


Figure 2: The Innovation Engine: Human culture creates amazing inventions, such as the telephone, by accumulating a multitude of interesting innovations in all directions. These stepping stones are collected, improved and then combined to create new innovations, which in turn, serve as the stepping stones for innovations in later generations. We propose to automate this process by having stochastic optimization (e.g. evolutionary algorithms) generate candidate solutions from the current archive of stepping stones and Deep Neural Networks evaluate whether they are *interestingly new* and should thus be archived.

inventions in many different domains, from math and science to art and engineering.

2.1 The ultimate goal

Our long-term vision is to create an Innovation Engine that does not require labeled data, or perhaps is not even shown data from the natural or man-made world. It would learn to classify the types of things it has produced so far and seeks to produce new types of things. Technically, one way to implement this algorithm is by training generative deep neural network models with unsupervised learning algorithms: these generative models can learn to compress the types of data they have seen before (Bengio, 2009; Hinton and Salakhutdinov, 2006). One could thus measure if a newly generated thing is a new type of thing by how well the generative DNN model can compress it. Evolution will be rewarded for producing things that the DNN cannot compress well, which should endlessly produce novel types of things.

Imagine such an Innovation Engine in the image domain. A network trained on all images produced so far will attempt to compress each newly generated image, and it will fail more on new types of images. We hypothesize that the DNN will continuously become "bored" with (i.e. highly compress) easily produced classes of images (initially static and solid colors, but soon more complex patterns), which will encourage evolution to generate increasingly complex images in order to produce new types of images. The process thus becomes a coevolutionary *innovation arms race*.

This version of the Innovation Engine is motivated by Schmidhuber et. al. curiosity works (Schmidhuber, 2006; Kompella et al., 2015) – which emphasizes the production of things that are not compressed yet, but are most easily compressed next – but our work involves modern compressors (state-of-the-art DNNs) and our algorithm does not require the seemingly impossible task of *predicting* which classes of artifacts are highly compressible. Our proposal is similar to (Liapis et al., 2013), but prevents cycling by attempting to produce things different than everything produced so far, not just the current population. If it works, this Innovation Engine could produce innovations in the multitude of fields and problem domains that currently benefit from stochastic optimization.

2.2 Version 1.0

Unsupervised learning algorithms for generative models do not yet scale well to high dimensional data (Bengio et al., 2014); for example, they can handle 28×28 pixel MNIST images (Hinton and Salakhutdinov, 2006) but not 256×256 pixel ImageNet images (Deng et al., 2009). In this section we describe a simpler Innovation Engine version that can be implemented with currently available algorithms. A key piece of the ultimate Innovation Engine is automatically recognizing new types of classes, which function as newly created niches for evolution to specialize on. We can emulate that endless process of niche creation by simply starting with a lot of niches and letting evolution exploit them all. To do that, we can take advantage of two recent developments in machine learning: (1) the availability of large, supervised datasets, and (2) the ability of modern supervised Deep Learning algorithms to train DNNs to reach near-human-competitive levels in classifying the things in these datasets (Hinton and Salakhutdinov, 2006; Krizhevsky et al., 2012; Bengio, 2009). We can thus challenge optimization algorithms (e.g. evolution) to produce things that the DNN recognizes as belonging to each class.

Innovation Engines require two key components: (1) a diversity-promoting EA that generates and collects novel behaviors, and (2) a DNN capable of evaluating the behaviors to determine if they are interesting and should be retained. The first criterion could be fulfilled either by Novelty Search or the multi-dimensional archive of phenotypic elites (MAP-Elites) algorithm (Mouret and Clune, 2015; Cully et al., 2015). We show below that both can work.

3 Test Domain: Generating Images

The test domain for the paper is generating a diverse set of interesting, recognizable images. We chose this domain for four reasons. The first is because an experiment in image generation served as the inspiration for Novelty Search (Stanley and Lehman, 2015). That experiment occurred on Picbreeder.org, a website that allowed visitors to interactively evolve images (Secretan et al., 2011), resulting in a crowd of humans that evolved a diverse, recognizable set of images. Key enablers of this diversity were (Secretan et al., 2011; Stanley and Lehman, 2015): the fact that collectively there was no goal; that individuals periodically had a target image type in mind, creating a local pressure for high-performing (recognizable) images; users were open to the possibility of switching to a new goal if the opportunity presented itself (e.g. if the eyes of a face started to look like the wheels of a car); that users saved any image that they found interesting (usually a new type of image, or an improvement upon a previous type of image) and future users could branch off of any saved stepping stone to create a new image. Critically, all of these elements should also occur in Innovation Engine 1.0; thus one test of that hypothesis is whether Innovation Engine 1.0 can automatically produce a diverse set of images like those generated by humans on Picbreeder. One attempt was made to automatically recreate the diversity of recognizable images produced on Picbreeder, but it produced only abstract patterns (Auerbach, 2012).

The second motivation for the image-generating domain is that DNNs are nearly human-competitive at recognizing images (Krizhevsky et al., 2012; Karpathy, 2014; Szegedy et al., 2015; Stallkamp et al., 2012). The third reason is that DNNs can recognize and sensibly classify the type of images from Picbreeder (Fig. 3), specifically images encoded by compositional pattern producing networks (CPPNs) (Stanley, 2007). We also encode images with CPPNs in our experiments (described below). The fourth reason is because humans are natural pattern recognizers, making us quickly and intuitively

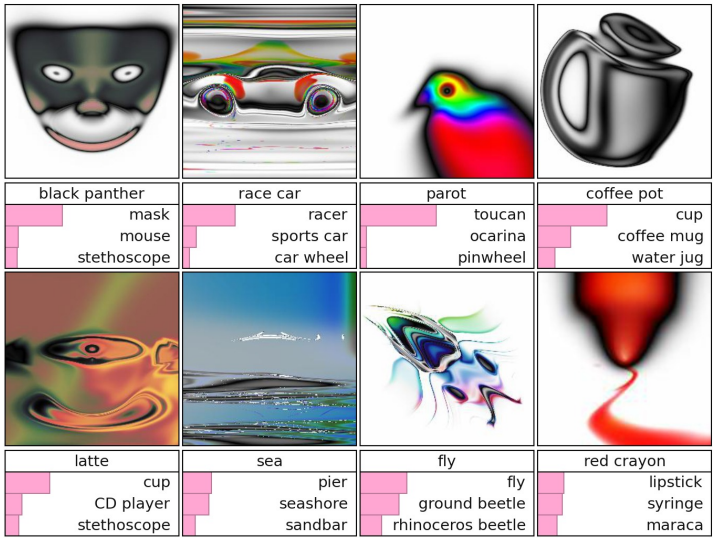


Figure 3: CPPN-encoded images evolved and named (centered text) by Picbreeder.org users. The DNN's top three classifications and associated confidence (size of the pink bar) are shown. The DNN's classifications often relate to the human breeder's label, showing that DNNs can recognize CPPN-encoded, evolved images. Adapted from (Nguyen et al., 2015a).

able to evaluate the diversity, interestingness, and recognizability of evolved solutions. Additionally, while much of what we learn from this domain comes from subjective results, there is also a quantitative aspect regarding the confidence a DNN ascribes to the generated images. In future work we will test whether the conclusions reached in this mostly subjective domain translate into more exclusively quantitative domains.

To experiment in this domain, we use a modern off-the-shelf DNN trained with 1.3 million images to recognize 1000 different types of objects from the natural world. We then challenge evolution to produce images that the DNN confidently labels as members of each of the 1000 classes. Evolution is therefore challenged to make increasingly recognizable images for all 1000 classes. Generating CPPN-encoded images that are recognizable is challenging (Woolley and Stanley, 2011), making recognizability a notion of performance in this domain. Being recognizable is also related to being interesting, as Picbreeder images that are recognizable are often the most highly rated (Secretan et al., 2011).

4 Methods

4.1 Deep neural network models

The DNN in our experiments is the well-known convolutional "AlexNet" architecture from (Krizhevsky et al., 2012). It is trained on the 1.3-million-image 2012 ImageNet dataset (Deng et al., 2009; Russakovsky et al., 2015), and available for download via the Caffe software package (Jia et al., 2014). The Caffe-provided AlexNet has small architectural differences from Krizhevsky 2012 (Krizhevsky et al., 2012), but it performs similarly (42.6% top-1 error rate vs. the original 40.7% (Krizhevsky et al., 2012)). For

each image, the DNN outputs a post-softmax, 1000-dimensional vector reporting the probability that the image belongs to each ImageNet class. The softmax means that to produce a high confidence value for one class, all the others must be low.

4.2 Generating images with evolution

To simultaneously evolve images that match all 1000 ImageNet classes, we use the new multi-dimensional archive of phenotypic elites (MAP-Elites) algorithm (Mouret and Clune, 2015; Cully et al., 2015). MAP-Elites keeps a map (archive) of the best individuals found so far for each class. Each iteration, an individual is randomly chosen from the map, mutated, and then it replaces the current champion for any class if it has a higher fitness for that class. Fitness is the DNN's confidence that an image is a member of that class.

We also test another implementation of the Innovation Engine, but with Novelty Search instead of MAP-Elites. Novelty Search encourages organisms to be different from the current population and an archive of previously novel individuals. The behavioral distance between two images is defined as the Euclidean distance between the two 1000-dimensional vectors output by the DNN for each image. Because all of our experiments were performed with the Sferes evolutionary computation framework (Mouret and Doncieux, 2010), we set all Novelty Search parameters to those in (Mouret, 2011), which was also conducted in Sferes, but followed closely the parameters in (Lehman and Stanley, 2008).

Images are encoded with compositional pattern producing networks (CPPNs) (Stanley, 2007), which abstract the expressive power of developmental biology to produce regular patterns (e.g. those with symmetry or repetition). CPPNs encode the complex, regular, recognizable images on Picbreeder.org (e.g. Fig. 3) and the 3D objects on EndlessForms.com (Clune and Lipson, 2011). The details of how CPPNs encode images and are evolved have been repeatedly described elsewhere (Secretan et al., 2011; Stanley, 2007). Briefly, a CPPN is like a neural network, but each node's activation function is one of a set (here: sine, sigmoid, Gaussian and linear). The Cartesian coordinates of each pixel are input into the network and the network's outputs determine the color of that pixel. Importantly, evolved CPPN images can be recognized by the DNN (Fig. 3), showing that evolution can produce CPPN images that both humans and DNNs can recognize.

As is customary (Secretan et al., 2011; Stanley, 2007; Clune and Lipson, 2011) we evolve CPPNs with the principles of the NeuroEvolution of Augmenting Topologies (NEAT) algorithm (Stanley and Miikkulainen, 2002), a version of which is provided in Sferes. CPPNs start with no hidden nodes, and add nodes and connections over time, forcing evolution to first search for simple, regular images before increasing complexity (Stanley and Miikkulainen, 2002). All of our code and parameters are available at `http://EvolvingAI.org`. Because each run required 128 CPU cores running continuously for ~4 days, our number of runs is limited.

5 Results

We conduct a variety of experiments to investigate Innovation Engines. First, we show that Innovation Engines work well both quantitatively and qualitatively in the image generation domain (Sec. 5.1): the algorithm produces images that are recognizable to both humans and DNNs. Second, we investigate a key component of Innovation Engines—the number of objectives—and show that the performance and evolvability improves as the number of objectives increases (Sec. 5.2). Third, to support the hypothesis that Innovation Engines should work with any diversity-promoting EA, we demonstrate that Innovation Engines also work well with Novelty Search (Sec. 5.3). Fourth, we show that the algorithm can be further improved by incorporating additional priors (Sec. 5.4).

5.1 Evolving images that are recognizable as members of ImageNet classes

If the Innovation Engine is a promising idea, then Innovation Engine 1.0 in the image domain should produce the following: (1) images that the DNN gives high confidence to as belonging to ImageNet classes and (2) a diverse set of interesting images that are recognizable as members of ImageNet classes. Our results show that the Innovation Engine accomplishes both of these goals.

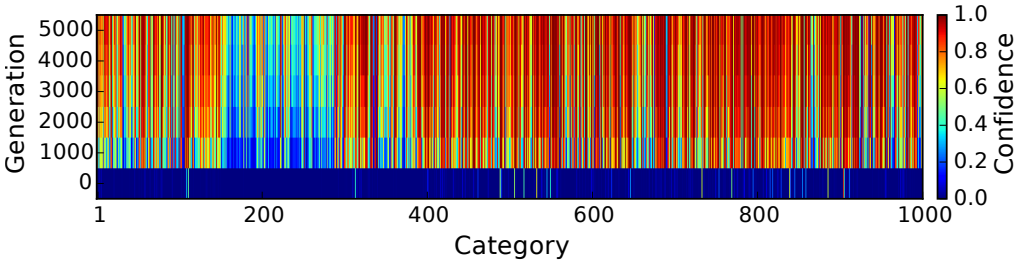


Figure 4: The MAP-Elites evolutionary algorithm produces images that the DNN declares with high confidence to belong to most ImageNet classes. Colors represent median confidence scores from 10 runs.

In 10 independent MAP-Elites runs, evolution produced high-confidence images in most of the 1000 ImageNet categories (Fig. 4). It struggles most in classes 156-286, which represent subtly different breeds of dogs and cats, where it is hard to look like one type without also looking like other types. Note that because the confidence values are taken after a softmax transformation of the neural activations of the last layer, to maximize its score in one class, an image not only has to have a high-confidence in that class, but also has to have a low-confidence in all the other classes; that is especially difficult for the dog and cat classes given the number of similar cat and dog breeds. While the reader must draw their own conclusions, in our opinion the images exhibit a tremendous amount of interesting diversity, putting aside whether they are recognizable. Selected examples are in Figs. 5, 1, and 6: all 10,000 evolved images are shown at http://www.evolvingai.org/InnovationEngine. The diversity is especially noteworthy because many images are phylogenetically related, which should curtail diversity.

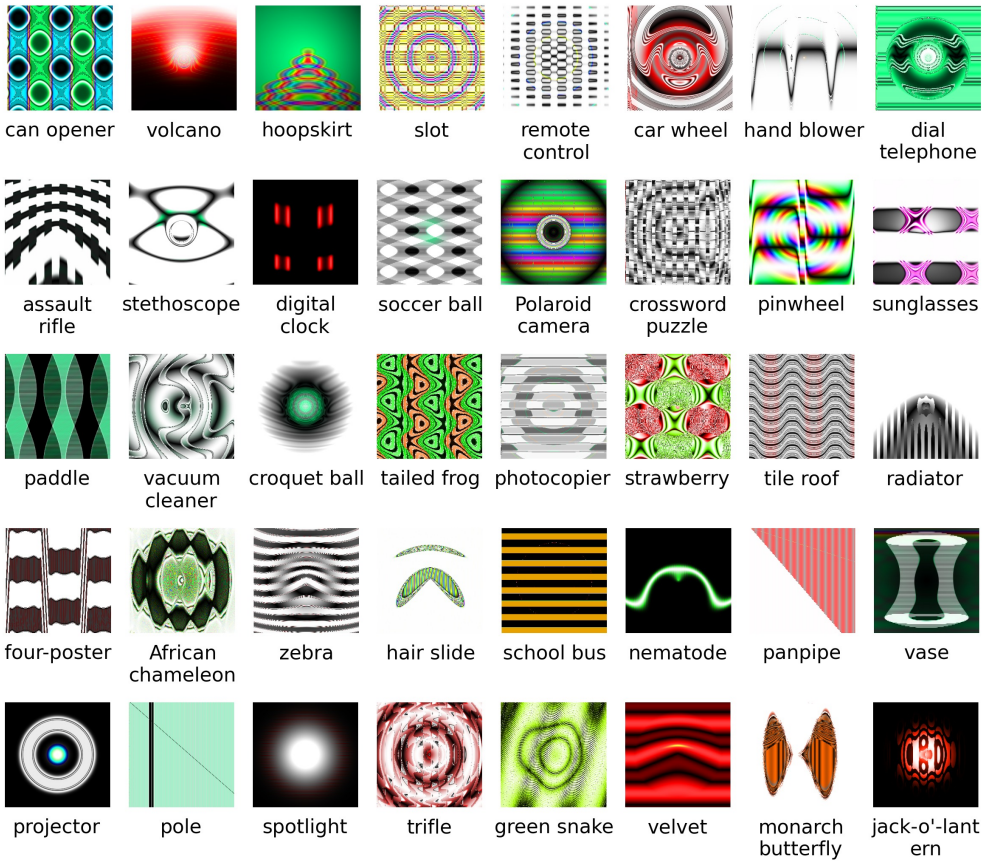


Figure 5: Innovation Engines in the image domain generate a tremendous diversity of interesting images. Shown are images selected to showcase diversity from 10 evolutionary runs. The diversity results from the pressure to match 1000 different ImageNet classes. In this and subsequent figures, the DNN's top label for each evolved image is shown below it.

In many cases, the evolved images are recognizable as members of the target class (Fig. 6). This result is remarkable given that it has been shown that with the same encoding (CPPN) and evolutionary algorithm (NEAT), it is impossible to evolve an image to resemble a complex, target image (Woolley and Stanley, 2011). The lesson from that paper is that if evolution is given a specific objective, such as to evolve a butterfly or skull, that it will not succeed because objective-driven evolution only rewards images that increasingly look like butterflies or skulls, and that CPPN lineages that lead to butterflies or skulls tend to pass through images that look nothing like either. Innovation Engines, like crowds on Picbreeder, simultaneously collect improvements in a large number of objectives. That allows evolutionary lineages to be rewarded for steps that do not resemble butterflies or skulls (provided they resemble something else) and then to be rewarded as butterflies or skulls if they subsequently resemble either. Thus, a main result of this paper is that the problem with traditional stochastic optimization is not that it is objective-driven, as is often argued (Lehman and Stanley, 2008, 2011a;

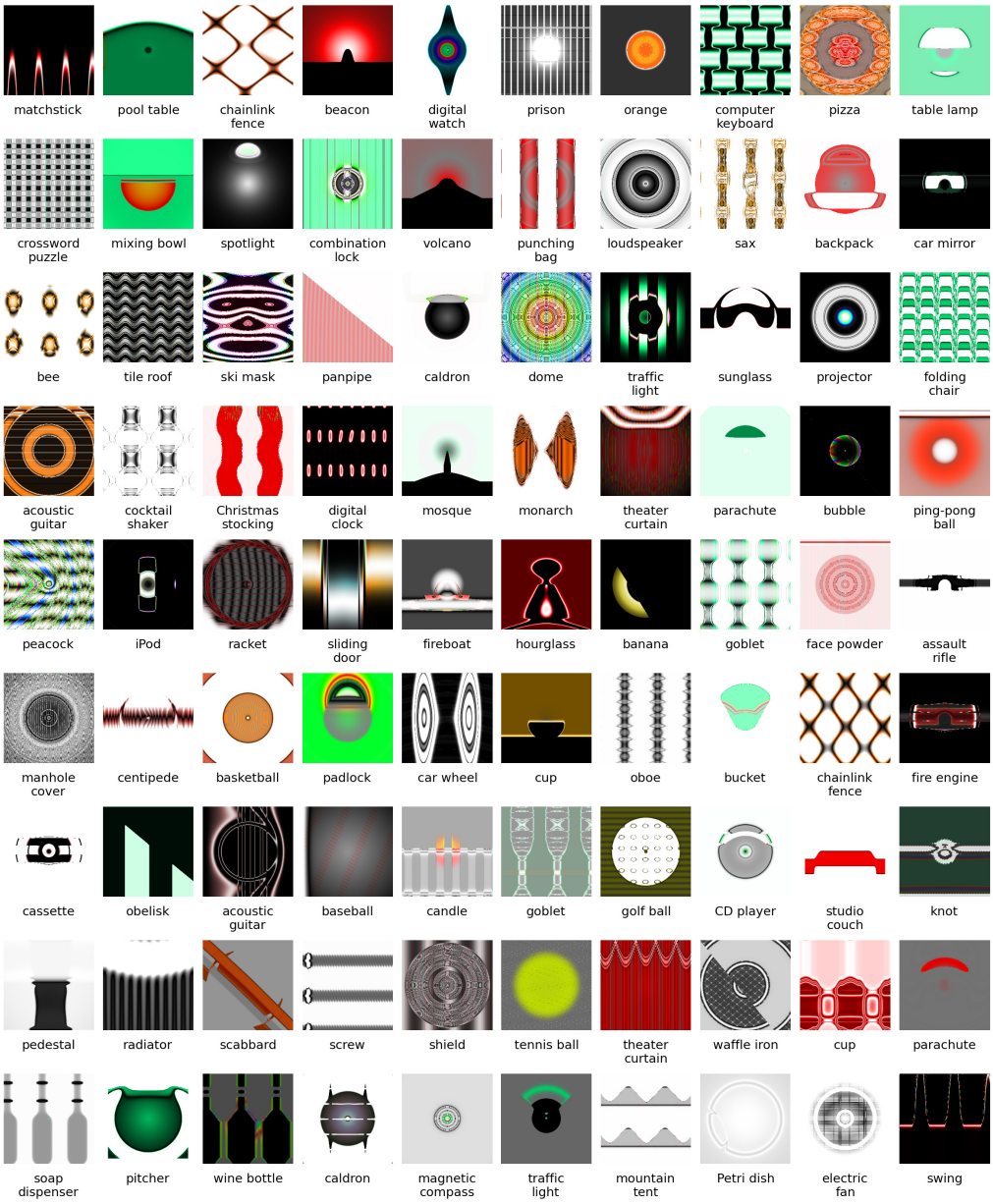


Figure 6: Innovation Engines are capable of producing images that are not only given high confidence scores by a deep neural network, but are also qualitatively interesting and recognizable. To show the most interesting evolved images, we selected images from both the 10 main experiment runs and 10 preliminary experiments that had slightly different parameters.

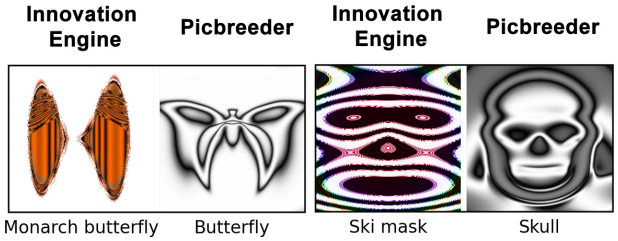


Figure 7: The Innovation Engine 1.0 evolved images that resemble those originally evolved on Picbreeder, but that a previous paper (Woolley and Stanley, 2011) showed were impossible to re-evolve with single-objective, target evolution. ImageNet has a "Monarch butterfly" class; it does not have a "skull" class, but its "Ski mask" class contains the key eyes, nose and mouth features. For each pair, the images shown are evolved with an Innovation Engine 1.0 (*left*) and Picbreeder (*right*).

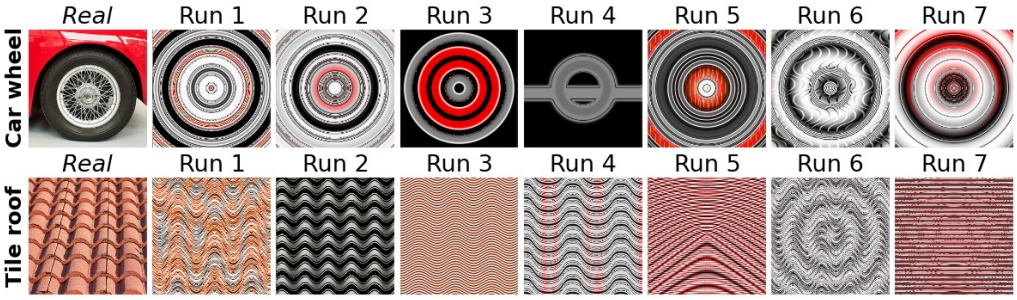


Figure 8: Final images evolved for the *Car wheel* and *Tile roof* classes from 7 independent runs. Common features – here, circles (top) and waves (bottom) – tend to show up consistently over different runs. The left column of images are from the ImageNet training set.

Stanley and Lehman, 2015), but instead that it is driven by only a few objectives. The key is to collect "stepping stones in all interesting directions", which can be approximated by simultaneously selecting for a vast number of objectives. Supporting this argument, our algorithm was able to produce many complex structures (Figs. 5, 1, 6) , including some that are similar to butterflies and skulls (Fig. 7).

We also qualitatively observed common features shared between the images evolved for the same target class over multiple runs of evolution (Fig. 8). For example, the *Car wheel* images tend to have concentric circles representing the tire and the rim. Images in *Tile roof* category tend to exhibit brownish terra-cotta color and the wavy pattern of the roof. This result shows that for certain categories, Innovation Engines can consistently produce images that are interesting and recognizable to both humans and DNNs.

Some evolved images are not recognizable, but often do contain recognizable features of the target class. For example, in Fig. 5, the *remote control* has a grid of buttons and the *zebra* has black-and-white stripes. As was recently reported in our paper titled "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images" (Nguyen et al., 2015a), this algorithm also produces many images that DNNs assign high confidence scores to, but that are totally unrecognizable, even when

Figure 9: Confirming that images evolved with Innovation Engines can be considered art, they were not only accepted to a selective art competition (35% acceptance rate) and displayed at the University of Wyoming Art Museum, but they also were amongst the 21% of submissions that won an award. Additionally, they will be displayed in art exhibits in galleries, fairs, and conventions in multiple European countries and the United States.

their class labels are known (e.g. Fig. 5, *tailed frog* & *soccer ball*). That study emphasized that the existence of such "fooling images" is problematic for anything that relies on DNNs to accurately classify objects, because DNNs sometimes make mistakes. This paper emphasizes the opposite, but not mutually exclusive, perspective: while using DNN as evaluators sometime produces fooling examples, it also sometimes works really well, and can thus automatically drive the evolution of a diverse set of complex, interesting, and sometimes recognizable images. Sec. 5.4 discusses a method for increasing the percent of evolved images that are recognizable.

To test the hypothesis that the CPPN images generated by Innovation Engines might actually be considered quality art, we submitted a selection of them to a selective art contest: the University of Wyoming's 40th Annual Juried Student Exhibition, which only accepted 35.5% of the submissions. Not only was the selection of Innovation Engine-produced images we submitted accepted, but it was also amongst the 21.3% of submissions to be given an award (Fig. 9).

5.2 Investigating whether having more objectives improves performance and evolvability

This section contains a number of experiments and analyses to probe a central hypothesis motivating Innovation Engines, which is that having more objectives will tend to improve performance (on each objective) and improve evolvability. We first present results and analyses from the "one class vs. 1000 classes" experiment that were reported in (Nguyen et al., 2015b), because they provide a nice illustration of the power of having many objectives. Then, in the subsequent section, we take a deeper dive into these questions, and do so across a range of objectives (1, 50, 100, 500, 1000) instead of comparing only 1 to 1000.

5.2.1 One objective vs. 1000 objectives

As discussed in the previous section, a key hypothesis for why Innovation Engines work is that evolving toward a vast number of objectives simultaneously is more effective than evolving toward each objective separately. In this section, we probe that hypothesis directly by comparing how MAP-Elites performs on all 1000 objectives vs. how evolution fares when evolving to each single-class objective separately. Because we did not have the computational resources to perform 1000 single-class runs, we randomly selected 100 classes from the ImageNet dataset and performed two single-class MAP-Elites runs per category. We compare those data to how the 10 runs of 1000-class MAP-Elites performed on the same 100-class subset.

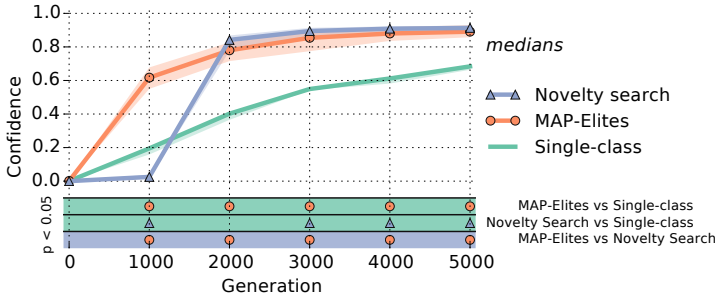


Figure 10: Innovation Engines built with MAP-Elites or Novelty Search perform similarly to each other, and both significantly outperform a single-class evolutionary algorithm. Solid lines show median performance and shaded areas indicate the 95% bootstrapped confidence interval of the median. The bottom three rows show statistical significance. For more information on the experiments that generated the data plotted here, see Sec. 5.2.1 for the MAP-Elites and single-class experiments, and Sec. 5.3 for the Novelty Search experiment.

1000-class MAP-Elites produced images with significantly higher median DNN confidence scores (Fig. 10, 90.3% vs. 68.3%, $p < 0.0001$ via Mann-Whitney U test). The theory behind why more objectives helps is because a lineage that is currently the champion in class X may be trapped on a local optima, such that mutations to it will not improve its fitness on that objective (a phenomenon we observe in the single-class case: Fig. 11 inset). With many objectives, however, a lineage that has been selected for other objectives can mutate to perform better on class X, which occurs frequently with MAP-Elites. For example, on the *water tower* class (Fig. 11 inset), the lineage of images that produce a large, top-lit sphere do not improve for 250 generations, but at generation 1750 a descendant of an organism that was the champion for the *cocker spaniel dog* class (Fig. 11) became a recognizable water tower and was then further refined.

Inspecting the phylogenetic tree of the 1000 images produced by MAP-Elites in each run, we found that the evolutionary path to a final image often went through other classes, a phenomenon we call *goal switching*. For example, the path to a *beacon* involved stepping stones that were rewarded because they were at one point champions for the *tench*, *abaya*, *megalith*, *clock*, and *cocker spaniel dog* classes (Fig. 11). A different descendant of *abaya* traversed the *stingray* and *boathouse* classes en route to a recognizable *planetarium* (Fig. 11). A related phenomenon occurs on Picbreeder, where the evolutionary path to a final image often involves images that do not resemble the target (Secretan et al., 2011).

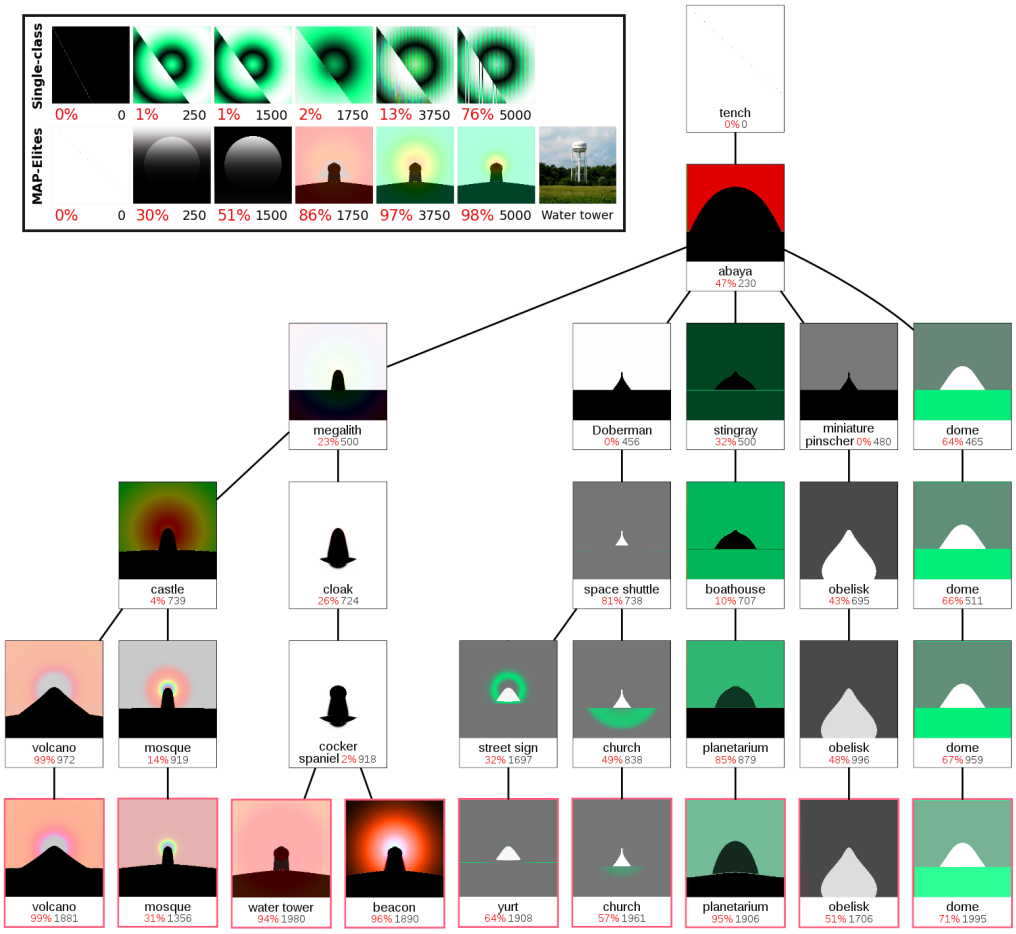


Figure 11: *Inset Panel:* The champions for the water tower class over evolutionary time for a single-class evolutionary algorithm (*top*) and the MAP-Elites variant of the Innovation Engine (*bottom*). Under each evolved image is the percent confidence the DNN has that the image is a water tower (left) and the generation in which the image was created (right). At the 1750th generation, when the offspring of a champion of the cocker spaniel (dog) class (see main panel in this figure) becomes the best water tower produced so far. Its descendants are refined to produce a high-confidence, recognizable image. *Main Figure:* A phylogenetic tree depicting how lineages evolve and *goal switch* from one class to another in an Innovation Engine (here, version 1.0 with MAP-Elites). Each image is displayed with the class the DNN placed it in, the associated DNN confidence score (*red*), and the generation in which it was created. Connections indicate ancestor-child relationships. One reason Innovation Engines work is because similar types of things (e.g. various building structures) can be produced by phylogenetically related genomes, meaning that the solution to one problem can be re-purposed for a similar type of problem. Note the visual similarity between the related solutions. Another reason Innovation Engines work is because the path to a solution often involves a series of things that do not increasingly resemble the final solution (at least, not without the benefit of hindsight). For example, note the many unrelated classes that served as stepping stones to recognizable objects (e.g. the path through cloaks and cocker spaniels to arrive at a beacon).

We quantitatively measured the number of goal switches per class (the number of times during a run that a new class champion was the offspring of a champion of another class). Each class had a $mean$ of 8.7 goal switches, which was 17.9% of the 48.6 mean new champions per class. Thus, a large percentage of improvements in a class came not from refining the current class champion, but from a mutation to a different class champion, helping to explain why Innovation Engines work.

Another expectation, which we observed, is that the evolved images for many semantically related categories are also phylogenetically related. For example, according to WordNet hierarchy (Deng et al., 2009), *planetarium*, *mosque*, *church*, *obelisk*, *yurt* and *beacon* are subclasses of the *structure* class (Fig. 12). The evolved images for these classes are often closely related phylogenetically and share visual similarities (Fig. 11).

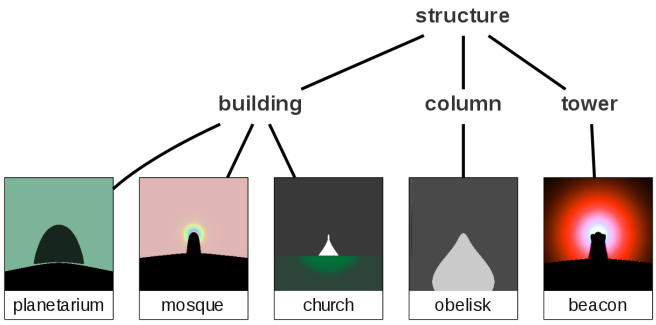


Figure 12: Evolved images sorted according to WordNet hierarchy. Planetarium, mosque, church, obelisk, and beacon images semantically belong to subclasses of the *structure* category. Interestingly, the images also exhibit similar visual patterns.

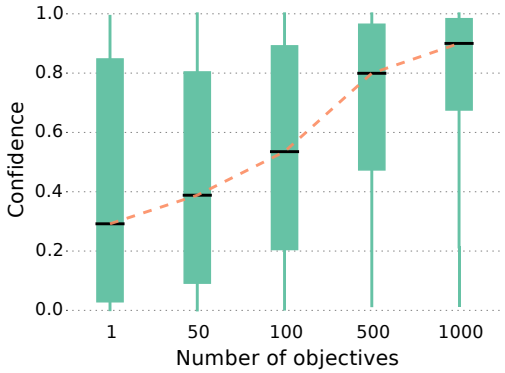
If two CPPN genomes produce equivalent behaviors (here, images), it is taken as a sign of increased evolvability if one has fewer nodes and connections (Lehman and Stanley, 2008; Woolley and Stanley, 2011; Secretan et al., 2011). It has been shown that objective-based search "corrupts" genomes by adding piecewise hacks that lead to small fitness gains, and thus do not find the simple, elegant solutions produced by divergent search processes (e.g. Novelty Search or Picbreeder crowds) (Woolley and Stanley, 2011). If Innovation Engines behave like traditional single- or multi-objective algorithms, one might expect them to produce large CPPN genomes. On the other hand, if Innovation Engines, which are *many*-objective algorithms, are more divergent in nature, they should produce smaller genomes like those reported for Picbreeder (Woolley and Stanley, 2011). While the comparison is not apples to apples for many reasons, Innovation Engine genomes are actually more compact than those for Picbreeder. The 10,000 MAP-Elites CPPN genomes contain a median of 27 nodes (SD $= 5.9$) and 37.5 connections (SD$= 8.6$) vs. the ~7,500 Picbreeder image genomes analyzed in (Secretan et al., 2011), which have 50.3 nodes and 146.7 connections (SD not reported).

5.2.2 Additional analyses and a more extensive sweep across the number of objectives

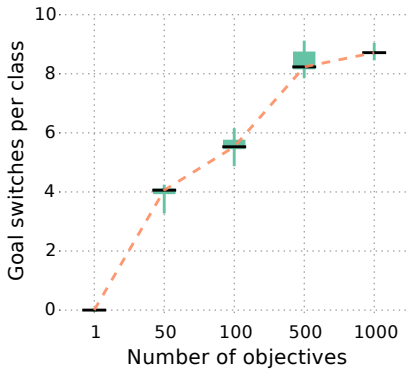
To further investigate how the number of objectives affects performance and evolvability, we conducted MAP-Elites experiments for a range of numbers of objectives: 1, 50, 100, 500, and 1000. In each, we restricted the MAP-Elites archive to keep a champion for N classes, where $N = 1, 50, 100, 500, 1000$. The classes are randomly selected from the 1000 ImageNet classes. In each generation, MAP-Elites produced 400 offspring by mutating a randomly selected champion from the set of N. Each of the 400 offspring would then be compared against every current class champion, and the offspring would replace a champion if its confidence score for that class was higher. For each treatment, the algorithmic hyperparameters were the same and we performed 10 independent runs.

Performance increases with the number of objectives

We found that the median performance increases monotonically as the number of objectives increases (Fig. 13a, $\rho = 0.24$, $p < 0.0001$ via Spearman's rank-order correlation). There are at least two potential explanations for this result. The first is our main hypothesis for why Innovation Engines work well, which is that increasing the number of objectives enables *goal switching* to occur more frequently (Fig. 13b). Supporting this explanation is the fact that the number of goal-switches also monotonically increases with the number of objectives ($\rho = 0.95$, $p < 0.0001$ via Spearman's rank-order correlation). A second possible explanation is that having fewer objectives results in less diversity among the 400 offspring in each generation, because those offspring descend from a smaller pool of parents. Having less diversity frequently hurts the performance of stochastic optimization algorithms, including evolutionary algorithms (Floreano and Mattiussi, 2008).



(a) Performance monotonically increases with the number of objectives.



(b) The number of goal switches monotonically increases with the number of objectives

Figure 13: As the number of objectives increases, both performance and the number of goal-switches per class also increase. This helps explain why evolving towards many objectives is effective. Orange lines show monotonically increasing relationships.

Are genomes more evolvable as the number of objectives increases?

As mentioned previously in section 5.2.1, it has previously been shown that increased evolvability in CPPN-encoded organisms can be detected by fewer nodes in CPPN genomes (Lehman and Stanley, 2008; Woolley and Stanley, 2011; Secretan et al., 2011). We showed in that section that the CPPNs evolved by Innovation Engines had fewer nodes and connections than the CPPNs from Picbreeder, although the comparison is not apples to apples. Here we test a related hypothesis: that having more objectives leads to CPPNs having fewer nodes and connections, which would suggest that they have more compact, elegant, evolvable representations.

Contrary to our hypothesis, the number of nodes and connections slightly, but significantly, increases with the number of objectives (Fig. 14, $\rho_{connections} = 0.08$, $\rho_{nodes} = 0.07$, $p < 0.0001$ via Spearman's rank-order correlation): The difference in the median number of nodes and connections between the 1-objective and the 1000-objective treatment is only 5 nodes and 5.5 connections, out of 27.7 nodes and 37.7 connections total on average. A study of Picbreeder genomes (Secretan et al., 2011) found a similar result that the size of CPPNs only slightly correlates with the human ratings for the evolved images ($\rho_{connections} = 0.11$, $\rho_{nodes} = 0.12$).

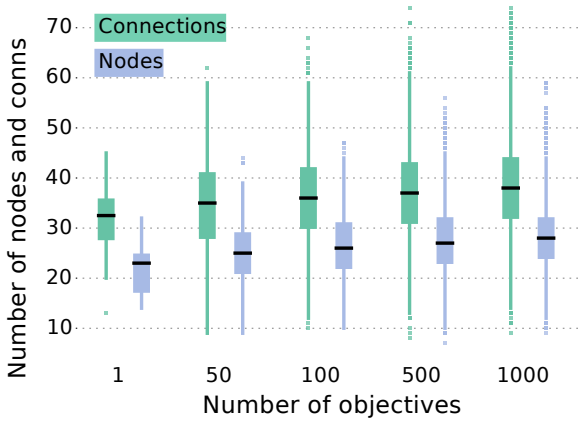
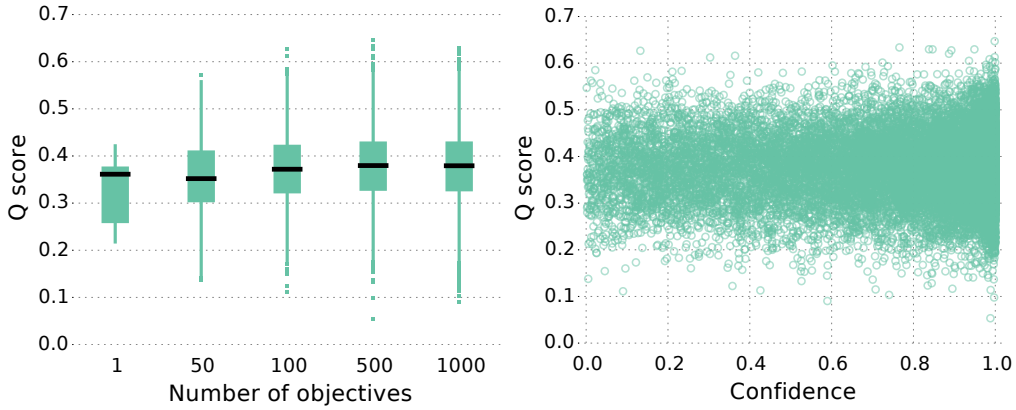


Figure 14: The size of CPPNs, in terms of nodes (*blue*) and connections (*green*), slightly, but significantly (see text) increases with the number of objectives.

A second indicator of evolvability is the *modularity* of genomes (Clune et al., 2013), because organisms designed in a modular fashion have been shown to adapt to new environments faster than non-modular organisms (Kashtan and Alon, 2005; Kashtan et al., 2007; Clune et al., 2013). Here, we test whether having a larger number of objectives produces CPPN genomes that are more modular. The structural modularity of CPPNs is measured by calculating its *Q score*, which is the most commonly used modularity metric for networks (Newman, 2006). Specifically, we treat each CPPN genome as a directed graph, and adopt the Q score metric for directed networks from (Leicht and Newman, 2008). Q scores are calculated for all 16,510 champion CPPNs from all 10 runs of all 5 treatments, where each treatment had 1, 50, 100, 500, or 1000 objectives.

We found a significant, but very slight, monotonic relationship between the number of objectives and the Q score (Fig. 15a, $\rho = 0.02$, $p < 0.01$ via Spearman's rank-order correlation). The lack of a strong relationship could be because in this domain, genomic modularity is not beneficial. Supporting that theory is the fact that there is only a very



(a) (b)

Figure 15: (a) There is only a slight monotonic relationship between the number of objectives in a treatment and the Q score of the class-champion CPPNs that evolve in that treatment, although that relationship is significant (see text). (b) Similarly, there is only an extremely slight, but significant (see text), correlation between the Q score of a CPPN and the confidence of the image that CPPN generates. Data points represent the class-champion CPPNs from all runs of all 5 treatments.

slight correlation between the Q score of the genome of each image and the confidence score of that image, although that relationship is also significant (Fig. 15b, $\rho = 0.05$, $p < 0.0001$ via Spearman's rank-order correlation). Using the same Q score metric, we also found that CPPNs generated by the 1000-objective treatment are significantly more modular than CPPNs evolved on Picbreeder (Stanley et al., 2013, 2016) (Q score $= 0.38$ vs. 0.28, $p < 0.0001$ via Mann-Whitney U test). While the comparison is not apples to apples for many reasons, this result shows that our automated evolution can produce similarly elegant, modular solutions to human-assisted evolution.

The evolvability of an organism can also be measured as a function of the fitness distribution of its offspring, with a higher distribution indicating increased evolvability (Hornby et al., 2003; Clune et al., 2011; Fogel and Ghozeil, 1996; Grefenstette, 2014; Belew et al., 1995; Igel and Chellapilla, 1999; Igel and Kreutz, 1999). To test whether more objectives leads to increased evolvability under this measure, we compared the fitness values of parents and their offspring. One challenge when measuring evolvability in this way is that the distribution of offspring fitness values may depend on the fitness of the parent. Since we have already shown that having more objectives improves performance, we need to control for the fitness of the parent in this evolvability analysis. To do that, we select a set of 400 different champions with performance values semi-evenly distributed within $[0, 1]$, such that each treatment has approximately the same distribution of fitness values, thus controlling for the fitness of the parents across treatments. Specifically, we divide the confidence range $[0, 1]$ into 20 bins: $[0.00, 0.05), [0.05, 0.1), ..., [0.95, 1)$. For each bin, every treatment contributes the same number of organisms. This number varies from bin to bin; however, on average, each treatment has 20 organisms in each bin for comparison. It was possible to fulfill these constraints for all treatments except the single-objective treatment: we thus include only the four treatments that had more than one objective ($N = 50, 100, 500, 1000$).

There are two separate sets of objectives over which we could have measured the fitness of offspring. Offspring could be compared vs. their parents on the class of the parent only, or across all 1000 ImageNet classes. We suspected that single- and few-objective treatments would have offspring that did better on the class of their parent, because organisms in these treatments spend most of their evolutionary history attempting to keep average fitness high on one or a few objectives. We further predicted that many-objective treatments would have organisms more evolved for goal-switching, such that their average fitness across all 1000 objectives would be higher. While both can be considered a form of evolvability, the goal-switching form of evolvability is what is truly required to solve extremely challenging problems and to make progress on open-ended evolution (Lehman and Stanley, 2008; Stanley and Lehman, 2015).

To measure within-class fitness changes, we produced 10 mutants per champion (each champion is from a class C), and measured their DNN confidence score improvement *in class C (only)* relative to the champion (i.e. the parent). In total, 400 champions $\times 10 = 4000$ mutants were considered per treatment. As expected, we found a very slight, but significant, negative monotonic correlation between the fitness changes of offspring with respect to their parent class only and the number of objectives in a treatment (Fig. 16, $\rho = -0.08$, $p < 0.0001$ via Spearman's rank-order correlation). Overall, across all treatments, 94.8% of the offspring have a lower DNN confidence score than their parents, but treatments with more objectives had distributions with slightly lower medians. Additionally, variance in the mutant confidence change distributions decreases as the number of objectives increases (Fig. 16). An explanation is that fewer-objective treatments produce organisms with lower confidence scores than more-objective treatments (Fig. 13a), leaving more room for the mutations to improve (thus, the higher variance). In the previous paragraph we outline one hypothesis for why single-objective or few-objective treatments have higher fitness distributions for their parent's class: because they have not been evolved to goal-switch as much.

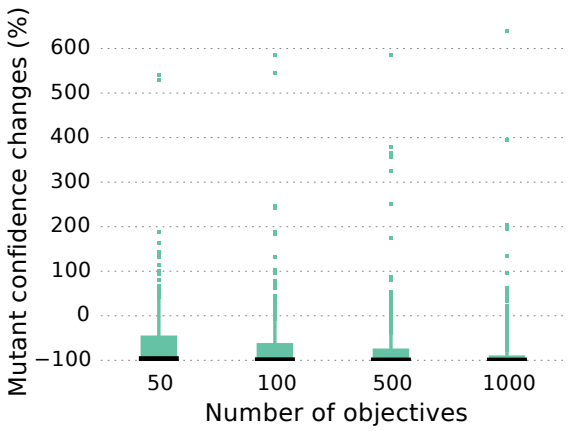


Figure 16: The median fitness changes of offspring compared to their parents (in percentage) for the same class as their parents slightly, but significantly decreases as the number of objectives increases (see text for statistics).

A perhaps better measure of evolvability is not just whether organisms fare better on the fitness peak their parents are on, but how they do across all fitness peaks. To

test this hypothesis, we not only need to control for the champion fitness, but also the number of classes that the champions came from. Specifically, we select 500 mutants (out of 4000 total) that have parents satisfying two conditions: 1) having confidence scores within the range $[0.95, 1)$; and 2) each coming from one of 50 randomly chosen classes. In order to have 500 champions that meet this criteria for all treatments, we were not able to constrain this set of 50 classes to be the same same for all treatments. We select from the 500 offspring produced per treatment the best image for each of the 1000 ImageNet classes. As expected, treatments with more objectives produce significantly higher average fitness across all 1000 classes (Fig. 17). This result confirms that the presence of multiple objectives leads to selection not just for high fitness, but for evolvability in the sense of being more likely to have a higher fitness on different objectives. This sort of evolvability could potentially aid our quest for open-ended evolutionary dynamics like those seen in nature (Lehman and Stanley, 2008; Stanley and Lehman, 2015).

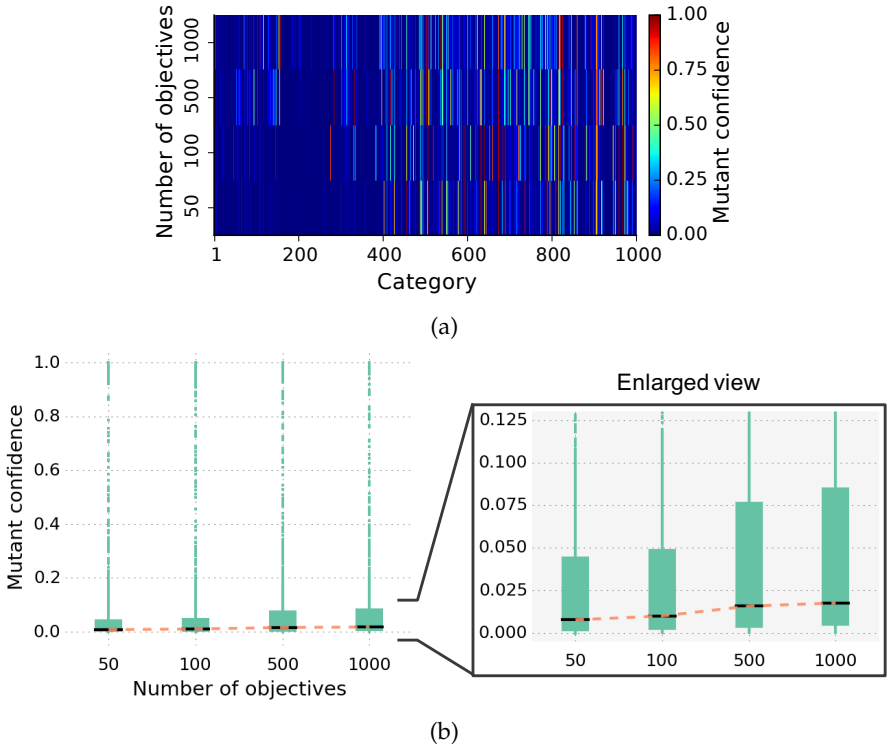


Figure 17: Organisms that are evolved with a higher number of objectives produce offspring that have higher average fitness across all 1000 objectives. (a) A fitness heatmap of 500 mutants for each treatment across 1000 categories. (b) The median performance of mutants across 1000 categories increases slightly, but significantly and consistently, with the number of objectives ($\rho = 0.14$, $p < 0.0001$ via Spearman's rank-order correlation). The inset panel (right) zooms in on the area of the left panel where the data is most concentrated to reveal the slight, but significant and monotonically increasing relationship between the number of objectives and evolvability.

Overall, the evidence is either neutral or positive supporting the claim that more objectives improves evolvability. While we did not find that more objectives leads to substantially higher modularity or genome compactness, there was a slightly positive, significant correlation between CPPN modularity and the number of objectives. More convincing is the fact that genomes evolved in the many objective environments are worse at staying high on the peak their parents are currently on, but have a significantly higher fitness distribution across all 1000 objectives. In other words, Innovation Engines are producing organisms with a form of evolvability that makes them more likely to goal-switch than EAs that have a single or low number of objectives.

5.3 Innovation Engine with Novelty Search

To support the case that Innovation Engines should work with any diversity-promoting EA combined with a DNN-provided *deep distance function*, we implemented Innovation Engine 1.0 with Novelty Search instead of MAP-Elites. After Novelty Search was afforded the same number of image evaluations, we found the best image it produced for each class according to the DNN. We performed 10 independent runs of Novelty Search. To facilitate comparison to the single-class control, we compare performance on the 100 classes randomly selected for the single-class control (Sec. 5.2). The MAP-Elites vs. Novelty Search comparison on 100 classes is qualitatively the same on all 1000 classes (data not shown).

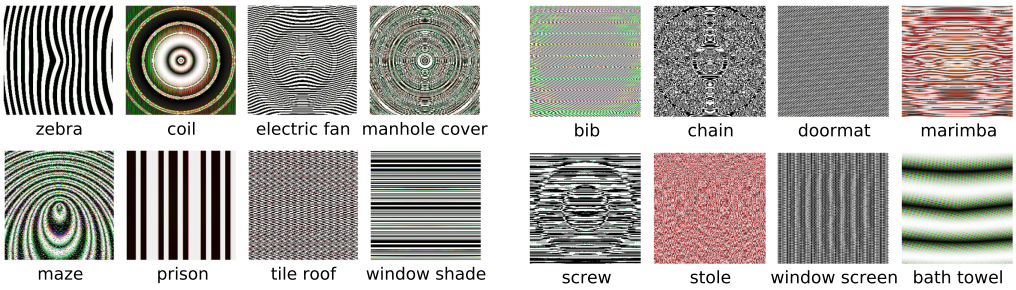
As expected, Novelty Search also produced high-confidence images in most classes (Fig. 10). Its median confidence of 91.6% significantly outperforms the 68.3% for the single-class control ($p < 0.0001$ via Mann-Whitney U test). While it significantly underperforms MAP-Elites at the 1000th generation, for the 2000th generation and beyond Novelty Search slightly, but significantly outperforms MAP-Elites ($p < 0.01$ via Mann-Whitney U test), although MAP-Elites has a higher final mean (79.5% vs. 74.0%). The images produced by two treatments are qualitatively similar (data not shown). This result confirms that in this domain both MAP-Elites and Novelty Search can serve as the diversity promoting EA in an Innovation Engine.

To isolate and evaluate the importance of the deep distance function in Innovation Engines, we launched 10 runs of the same Novelty Search experiment as above except that we replaced the DNN deep distance evaluation function with a pixel-by-pixel L_1 distance function. Specifically, the behavior distance D between two evolved images $\mathbf{u}$ and $\mathbf{v}$ of size 256×256 is calculated as the L_1 distance (aka Manhattan distance) between all pixel values across all 3 color channels $\mathbf{c}$:

$$D(\mathbf{u}, \mathbf{v}) = \sum_{c=1}^{3} \sum_{x=1}^{256} \sum_{y=1}^{256} |\mathbf{u}_{x,y}^{c} - \mathbf{v}_{x,y}^{c}| \quad (1)$$

This experiment, which swaps out the deep distance function with a shallow, pixel-wise distance function, was only performed with Novelty Search because it is not obvious how to sensibly discretize the space of all possible pixel combinations into bins, as MAP-Elites requires (Mouret and Clune, 2015; Cully et al., 2015).

The results reveal that Novelty Search with the L_1 distance function performs poorly: the images it produces are given extremely low-confidence scores by the DNN. After 2000 generations, the median confidence score is 0.18%, a significantly and substantially lower performance than the 84.4% for Novelty Search with the deep distance function ($p < 0.0001$ via Mann-Whitney U test). Since its performance is significantly lower than all other treatments at generation 2000 ($p < 0.0001$ via Mann-Whitney U



(a) Hand-selected images that we found recognizable. Median confidence: 94.25%

(b) A random sampling of images. Median confidence: 13.49%

Figure 18: Images produced by Novelty Search with a shallow, L_1 distance function directly in the pixel space. This experiment shows what happens when Novelty Search attempts to produce novel images without the deep distance function provided by a deep neural network, and instead encourages knowledge directly in the high-dimensional space of all possible pixel combinations. These results suggest that, in any domain, it is better to search along low-dimensional manifolds that represent interesting dimensions of variation, rather than searching directly in high-dimensional search spaces. Note: while the behavioral metric for Novelty Search is a pixel-wise distance function, the images are still generated by CPPNs, which explains why they are regular.

test), and because this experiment was computationally expensive due to the number of pixel comparisons that need to be made to calculate L_1 distances between images, we did not run the experiment all the way to 5000 generations, as we did for the other treatments (Fig. 10). While most images are uninteresting and unrecognizable (Fig. 18b), we found a few high-scoring images with recognizable patterns (Fig 18a, e.g. black-and-white stripes for an image in the *zebra* class and vertical bars for an image in the *prison* class). This experiment confirms that Novelty Search has difficulty finding the rare, interesting, recognizable images in the vast space of all possible pixel combinations. It suggests that, in general, Novelty Search will struggle to find interesting, rare items in a vast, high-dimensional space without a deep distance function that can focus the search on the interesting low-dimensional manifolds that exist with the higher dimensional space.

5.4 Attempting to further improve the frequency and quality of recognizable images by adding a natural image prior

Although DNNs sometimes make mistakes, giving high confidence scores to unrecognizable "fooling" images (Nguyen et al., 2015a), we recently showed that using a collection of *image priors* to bias optimization towards producing images with more natural image statistics can help produce more recognizable images (Yosinski et al., 2015). That finding was not for evolved images but for images produced via gradient-based optimization methods in which gradients are backpropagated to each pixel to search for images that maximally activate certain neurons in DNNs. We subsequently found that minimizing one particular prior called *total variation* (Rudin et al., 1992) produces even more recognizable images (Nguyen et al., 2016b). We hypothesized that encouraging the minimization of total variation in the fitness function via a penalty for higher total variation would encourage evolution to search for more regular images with constant color patches, and that this bias would thus improve recognizability, at least in the

directly encoded images.

First, we evolved images to match the MNIST handwritten digits dataset (LeCun et al., 1998b). For these experiments, the EA is the default MAP-Elites, as in our previous experiments, but the DNN is LeNet, which is a commonly used DNN for MNIST studies (LeCun et al., 1998a). We trained LeNet to successfully classify the 10 digits [0-9] of the MNIST dataset with an accuracy of 99.06%. Images are 28×28 large, grayscale, and directly encoded. Specifically, an image is encoded as a vector of integers, each representing the 8-bit grayscale value of a pixel. We then used the Innovation Engine to produce each image $\mathbf{u}$ in such a way that simultaneously a) maximizes the DNN confidence score $\Phi(\mathbf{u})$, and b) minimizes the total variation penalty $TV(\mathbf{u})$, via the following fitness function:

$$F(\mathbf{u}) = \Phi(\mathbf{u}) \times \big(\alpha - TV(\mathbf{u})\big) \tag{2}$$

The multiplicative fitness function here tries to encourage both $\Phi(\mathbf{u})$ and $\big(\alpha - TV(\mathbf{u})\big)$ terms to be high. Maximizing the first term means that the target neuron's activation should be high. Maximizing the second term means that TV should be minimized. We empirically chose $\alpha = 10^6$ as an upper-bound on total variation to make sure $\big(\alpha - TV(\mathbf{u})\big)$ is always positive.

Intuitively, the total variation of an image $\mathbf{u}$ is the sum of the variations in the color space between adjacent pixels (Rudin et al., 1992). Specifically, we use a common finite differences method for calculating the total variation norm (Getreuer, 2012), which is:

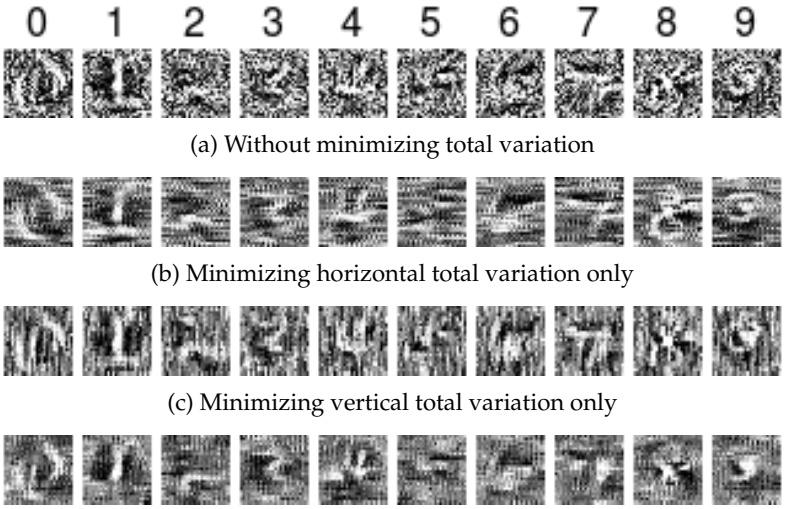
$$TV(\mathbf{u}) = \sum_{x,y} \sqrt{\left(\frac{\mathbf{u}_{x+1,y} - \mathbf{u}_{x-1,y}}{2}\right)^2 + \left(\frac{\mathbf{u}_{x,y+1} - \mathbf{u}_{x,y-1}}{2}\right)^2} \tag{3}$$

where (x, y) is the location of a pixel in the image. Note that since the variation of a pixel at (x, y) is measured with respect to two adjacent pixels, the variation is not calculated for pixels at the edge of an image (i.e. those on the outside border). Removing either the first or second term within the square root of Eq. 3 allows the smoothing effect to be applied along only the horizontal or vertical direction, respectively (Figs. 19b, 19c).

A recent study showed that it takes only 100 generations to produce unrecognizable images that the LeNet DNN classifies as digits with 99.99% confidence (Nguyen et al., 2015a). Here, we ran that experiment 50 times longer to 5000 generations with and without a penalty for total variation. Without a total variation penalty, these additional generations do enable evolution to produce recognizable images, at least for some classes, although the images contain lots of high-frequency noise static (Fig. 19a). Adding a total variation penalty to the fitness function generations produced smoother, more recognizable images with less noise (Figs. 19b, 19c, 19d). While the total variation prior is thus beneficial, its benefit only shows up after thousands of generations, making it extremely computationally expensive with only marginal benefit.

We also tested adding a total variation penalty on the challenge of producing images that resemble ImageNet classes, but it did not qualitatively improve image recognizability or DNN confidence scores after 20,000 generations (data not shown), although perhaps it would with much more computation than we had available.

When conducting the same experiment on CPPNs instead of directly encoded images, we found that total variation regularization does not improve the recognizability of images. An explanation for that is that CPPN-encoded images already tend to be smooth and regular (Fig. 6), and thus already have low total variation.



(a) Without minimizing total variation

(b) Minimizing horizontal total variation only

(c) Minimizing vertical total variation only

(d) Minimizing both horizontal and vertical total variation

Figure 19: Images evolved after 5000 generations with a direct encoding to match MNIST digit classes. Total variation regularization helps remove the static noise from the images, but requires thousands of generations to produce recognizable images. *(a)* Evolution is asked to maximize the DNN confidence score only. *(b, c, d)* Evolution is asked to both maximize the DNN confidence score, and minimize the total variation across an image.

All told, the total variation prior may help the directly encoded Innovation Engine in the image domain, but not enough to make a large qualitative difference. As expected, the total variation prior does not help with the already regular indirectly (CPPN) encoded images. To greatly improve the frequency and quality of the production of recognizable images, more research is needed to identify better priors that penalize non-natural images (Yosinski et al., 2015).

6 Discussion and Conclusion

This paper introduces the concept of the Innovation Engine. It then describes a version of Innovation Engines that can be created with existing deep learning technology, relying on DNNs trained with supervised learning. It also describes a more ambitious Innovation Engine that will take advantage of unsupervised learning technology once it is more mature. Our paper also also offers a first empirical investigation into many different aspects of Innovation Engines, including why they work and the degree to which they promote evolvability.

All of the work in this paper is in the domain of generating images. However, Innovation Engines should theoretically work in any domain, but future work is required to validate that hypothesis. In a future study, we will create Innovation Engines in more quantitative domains. For example, we will pair DNNs trained to recognize different actions in videos (e.g. cartwheels, backflips, handshakes) with evolutionary algorithms to attempt to automatically create neural network robot controllers for thousands of different robotic behaviors. DNNs already can classify the actions taking place in videos (Karpathy et al., 2014; Simonyan and Zisserman, 2014; Donahue et al., 2015),

and EAs can evolve neural networks to produce a variety of robot behaviors (Floreano and Mattiussi, 2008; Floreano and Keller, 2010; Cully et al., 2015; Clune et al., 2011; Li et al., 2014; Clune et al., 2009; Lee et al., 2013; Cheney et al., 2013), so we are optimistic that an Innovation Engine in this domain will be successful. That said, its computational costs will be substantial, given how expensive it is to both have DNNs evaluate videos and for EAs to simulate robot behaviors.

Our results have shown that the Innovation Engine concept is worth exploring further. Specifically, we have supported some of its key assumptions: that evolving toward many objectives simultaneously approximates divergent search; that DNNs can provide informative, abstract distance functions in high-dimensional spaces; and that Innovation Engines can generate a large, diverse, interesting set of solutions in a given domain (here images). Innovation Engines will only get better as DNNs are improved, especially when generative DNN models trained with unsupervised learning can scale to higher dimensions. Ultimately, Innovation Engines could potentially be applied to the countless number of domains where stochastic optimization is applied. Like human culture, they could eventually enable endless innovation in any domain, from software and science to arithmetic proofs and art.

Acknowledgements

We thank Joost Huizinga, Christopher Stanton, Henok Mengistu and Jean-Baptiste Mouret for useful conversations. Jeff Clune was supported by an NSF CAREER award (CAREER: 1453549) and a hardware donation from the NVIDIA Corporation and Jason Yosinski by the NASA Space Technology Research Fellowship and NSF grant 1527232.

References

Auerbach, J. E. (2012). Automated Evolution of Interesting Images. In *Artificial Life 13*. MIT Press.

Belew, R. K., Belew, R. K., and Vose, M. D. (1995). *Foundations of Genetic Algorithms*. Morgan Kaufmann.

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*.

Bengio, Y., Thibodeau-Laufer, É., Alain, G., and Yosinski, J. (2014). Deep generative stochastic networks trainable by backprop. In *Proceedings of the International Conference on Machine Learning*, pages 226–234.

Cheney, N., MacCurdy, R., Clune, J., and Lipson, H. (2013). Unshackling evolution: Evolving soft robots with multiple materials and a powerful generative encoding. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, GECCO '13, pages 167–174, New York, NY, USA. ACM.

Clune, J., Beckmann, B. E., Ofria, C., and Pennock, R. T. (2009). Evolving coordinated quadruped gaits with the hyperneat generative encoding. In *Proceedings of the 11th IEEE Congress on Evolutionary Computation*, CEC '09, pages 2764–2771, Piscataway, NJ, USA. IEEE Press.

Clune, J. and Lipson, H. (2011). Evolving 3d objects with a generative encoding inspired by developmental biology. *SIGEVOlution*, 5(4):2–12.

Clune, J., Mouret, J.-B., and Lipson, H. (2013). The evolutionary origins of modularity. *Proceedings of the Royal Society of London B: Biological Sciences*, 280(1755):20122863.

Clune, J., Stanley, K., Pennock, R., and Ofria, C. (2011). On the performance of indirect encoding across the continuum of regularity. *IEEE Transactions on Evolutionary Computation*, 15(4):346–367.

Cuccu, G. and Gomez, F. (2011). *When Novelty Is Not Enough*, pages 234–243. Springer Berlin Heidelberg, Berlin, Heidelberg.

Cully, A., Clune, J., Tarapore, D., and Mouret, J.-B. (2015). Robots that can adapt like animals. *Nature*, 521(7553):503–507.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '09, pages 248–255. IEEE.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '15, pages 427–436. IEEE.

Floreano, D. and Keller, L. (2010). Evolution of adaptive behaviour in robots by means of darwinian selection. *PLoS Biology*, 8(1):1–8.

Floreano, D. and Mattiussi, C. (2008). *Bio-inspired artificial intelligence: theories, methods, and technologies*. MIT press.

Fogel, D. B. and Ghozeil, A. (1996). Using fitness distributions to design more efficient evolutionary computations. In *Proceedings of the 1996 IEEE International Conference on Evolutionary Computation*, pages 11–19. IEEE.

Getreuer, P. (2012). Rudin-osher-fatemi total variation denoising using split bregman. *Image Processing On Line*, 2:74–95.

Grefenstette, J. J. (2014). Predictive models using fitness distributions of genetic operators. *Foundations of Genetic Algorithms*, 3:139–161.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

Hornby, G. S., Lipson, H., and Pollack, J. B. (2003). Generative representations for the automated design of modular physical robots. *IEEE Transactions on Robotics and Automation*, 19(4):703–719.

Igel, C. and Chellapilla, K. (1999). Fitness distributions: Tools for designing efficient evolutionary computations. *Advances in genetic programming*, 3:191–216.

Igel, C. and Kreutz, M. (1999). Using fitness distributions to improve the evolution of learning structures. In *Proceedings of the 1st IEEE Congress on Evolutionary Computation*, volume 3 of *CEC '99*, page 1909. IEEE Press.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, pages 675–678, New York, NY, USA. ACM.

Karpathy, A. (2014). What I learned from competing against a convnet on ImageNet. Retrieved from http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 1725–1732, Washington, DC, USA. IEEE.

Kashtan, N. and Alon, U. (2005). Spontaneous evolution of modularity and network motifs. *Proceedings of the National Academy of Sciences of the United States of America*, 102(39):13773–13778.

Kashtan, N., Noor, E., and Alon, U. (2007). Varying environments can speed up evolution. *Proceedings of the National Academy of Sciences*, 104(34):13711–13716.

Kompella, V. R., Stollenga, M., Luciw, M., and Schmidhuber, J. (2015). Continual curiosity-driven skill acquisition from high-dimensional video inputs for humanoid robots. *Artificial Intelligence*.

Koza, J. R., Keane, M. A., Streeter, M. J., Mydlowec, W., Yu, J., and Lanza, G. (2005). Genetic programming iv: Routine human-competitive machine intelligence. *Genetic Programming and Evolvable Machines*, 6:231–233.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998a). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

LeCun, Y., Cortes, C., and Burges, C. J. (1998b). The MNIST database of handwritten digits. Retrieved from http://yann.lecun.com/exdb/mnist/.

Lee, S., Yosinski, J., Glette, K., Lipson, H., and Clune, J. (2013). *Evolving Gaits for Physical Robots with the HyperNEAT Generative Encoding: The Benefits of Simulation*, pages 540–549. Springer Berlin Heidelberg, Berlin, Heidelberg.

Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In *Proceedings of the Eleventh International Conference on Artificial Life (Alife XI*. MIT Press.

Lehman, J. and Stanley, K. O. (2011a). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223.

Lehman, J. and Stanley, K. O. (2011b). Novelty search and the problem with objectives. In *Genetic Programming Theory and Practice IX*, pages 37–56. Springer.

Leicht, E. A. and Newman, M. E. (2008). Community structure in directed networks. *Physical review letters*, 100(11):118703.

Li, J., Storie, J., and Clune, J. (2014). Encouraging creative thinking in robots improves their ability to solve challenging problems. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 193–200, New York, NY, USA. ACM.

Liapis, A., Martınez, H. P., Togelius, J., and Yannakakis, G. N. (2013). Transforming exploratory creativity with delenox. In *Proceedings of the Fourth International Conference on Computational Creativity*.

Mouret, J.-B. (2011). *Novelty-Based Multiobjectivization*, pages 139–154. Springer Berlin Heidelberg, Berlin, Heidelberg.

Mouret, J.-B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*.

Mouret, J.-B. and Doncieux, S. (2010). Sferes v2: Evolvin'in the multi-core world. In *Proceedings of the 12th IEEE Congress on Evolutionary Computation*, CEC '10, pages 1–8. IEEE Press.

Newman, M. E. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582.

Nguyen, A., Dosovitskiy, A., Yosinski, J., Brox, T., and Clune, J. (2016a). Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *arXiv preprint arXiv:1605.09304*.

Nguyen, A., Yosinski, J., and Clune, J. (2015a). Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '15, pages 427–436. IEEE.

Nguyen, A., Yosinski, J., and Clune, J. (2015b). Innovation engines: Automated creativity and improved stochastic optimization via deep learning. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, GECCO '15, pages 959–966. ACM.

Nguyen, A., Yosinski, J., and Clune, J. (2016b). Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. In *Visualization Workshop, International Conference on Machine Learning (ICML)*.

Rudin, L. I., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Schmidhuber, J. (2006). Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. *Connection Science*, 18(2):173–187.

Secretan et al., J. (2011). Picbreeder: A case study in collaborative evolutionary exploration of design space. *Evolutionary Computation*, 19(3):373–403.

Simonyan, K. and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27*, pages 568–576. Curran Associates, Inc.

Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332.

Stanley, K., Huizinga, J., and Clune, J. (2016). The emergence of canalization and evolvability in an open-ended, interactive evolutionary system. *In Preparation*.

Stanley, K. and Lehman, J. (2015). *Why Greatness Cannot Be Planned: The Myth of the Objective*. Springer.

Stanley, K. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127.

Stanley, K. O. (2007). Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines*, 8(2):131–162.

Stanley, K. O., Clune, J., D'Ambrosio, D. B., Green, C. D., Lehman, J., Morse, G., Pugh, J. K., Risi, S., and Szerlip, P. (2013). CPPNs effectively encode fracture: A response to critical factors in the performance of hyperneat. Technical Report CS-TR-13-05, University of Central Florida.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '15, pages 1–9. IEEE.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014). Intriguing properties of neural networks. In *International Conference on Learning Representations*.

Woolley, B. G. and Stanley, K. O. (2011). On the deleterious effects of a priori objectives on evolution and representation. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, GECCO '11, pages 957–964, New York, NY, USA. ACM.

Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., and Lipson, H. (2015). Understanding neural networks through deep visualization. In *Deep Learning Workshop, International Conference on Machine Learning (ICML)*.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Proceedings of the 13th European Conference on Computer Vision (ECCV)*, pages 818–833, Cham. Springer International Publishing.