Introspective Neural Networks

Sam Powers
snpowers@cs.cmu.edu

Leonid Keselman
lkeselma@cs.cmu.edu

Abstract

We demonstrate that pre-trained neural networks contain meta-information that can be used to improve fine-grained recognition. This is demonstrated via two different experiments. Our first set of experiments shows that style-transfer methods can be used to enhance images, either enhancing the salient features of their class or shifting them to another desired class. Our second set of results shows that looking at the activations of pre-trained neural networks can provide information about novel classes. The first method is a progression towards using good top-5 classification accuracy to provide good top-1 accuracy. The second method builds towards online classifiers that expand their class distributions over time.

1. Introduction

Pre-trained neural networks models are currently the standard baseline model for tasks in computer vision. These networks contain some properties and features that make them useful for tasks in general [9]. These models seem to capture perceptual and semantic properties of images at a variety of scales, and even transfer these properties to other images in a believable way [3].

Our overall project goal is to explore about the extent to which information about the expected state of the world is contained within pre-trained weights. We'll tackle this broad goal from two angles. As shown in Fig 1, the first is to improve disambiguation between classes by actively leveraging the features the network expects to see. We initially tried to use feature inversions, but style transfer was a more successful approach. The second angle, as shown in Fig 2 is to use the network to detect when observation violates expectation, which could indicate data from an unseen class or an adversary. Both of these methods fit in the general theme of exploiting internal properties of pre-trained neural networks to further improve performance for well-studied problem of fine-grain classification. Since there are internal properties, we call the project introspective neural networks.

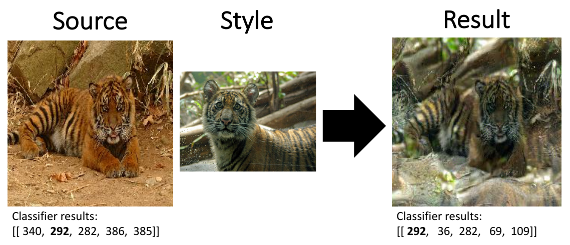

Figure 1. A demonstration of our first method. This utilizes style transfer [3] between exemplars. The source image is a tiger image on which a pre-trained classifier gets a correct top-5 result, but the correct label (tiger) is second (to the label zebra). After taking a known tiger exemplar picture as a style image, we obtain a result from style transfer where the same pre-trained network gets the correct top-1 prediction. Our experiments have shown that style transfer can be optimized to perform such class transformation effectively.

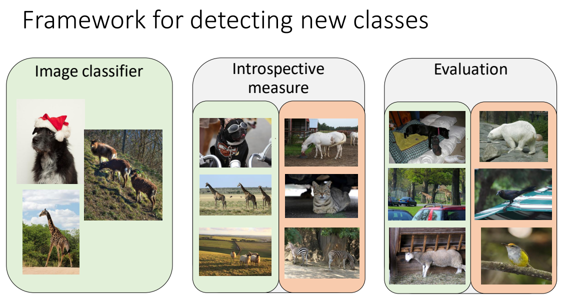

Figure 2. A demonstration of our second method. Initially we build a classifier over some set of classes (e.g. dog, giraffe, sheep). We then train a binary classifier between those classes and some new set (e.g. cat, horse, zebra). Finally, we evaluate on a dataset that has the original classes (dog, giraffe, sheep) and a different novel set (e.g. bear, crow, finch). All three steps of the process (fine-tuning, training a binary classifier, and evaluation) use different instances for all classes. The partitioning of classes used during training of the novel class detector.

2. Related Work

2.1. Fine Grained Classification

Neural networks are able to accentuate class details through *inversions* [14]. By utilizing this capability of enhancing a class's detail, we're interested in whether we're able to learn more fine grained classification models. One technical approach is to train a GAN [4] to detect real images from fake images, enhance/invert neural network features to emphasize all top 5 classes, and hope that our GAN selects which is most plausible as a real image. Similar ideas and models have been explored for super-resolution [13], but not for enhancing performance. We originally attempted to use a GAN but found them difficult to train and moved onto using style-transfer methods [3], which have a more visually appealing result. Style-transfer work has been shown to represent perceptual loss [11], for making images capture desired perceptual detail.

2.2. Novel Classes

Another interesting area is understanding the statistics and behavior of pre-trained neural network activations and weights. For example, can neural network statistics tell us if we're seeing data from a new distribution of images? For this, we follow adversarial perturbation and defense literature. For example, one can train a classifier based on global activation statistics, as well as neurons of interest [15]. These statistics can detect adversarial examples [5], although recent work has found weaknesses in those statistical approaches [1].

Many works focus on looking at network statistics during training time, for example gradient-flow [20] can be used to weight features during fine-tuning, preferring to only updates weights which lead to good localization.

3. Methods

3.1. Style transfer for enhancing class details

To the cause of improving top-1 accuracy given top-5 results, we desire to build a function that can shift an image to each of the top-5 classes. For our experiments, we build on style-transfer methods [3]. Style transfer is accomplished by extracting the activations of several layers of a pre-trained neural network (in our case, we use SqueezeNet [10] pre-trained on ImageNet). The Gram matrix of each feature layer is then formed. The Gram matrix for a layer l, G^l, is a matrix representing feature correlations, where given a feature map F, G^l_{ij} is defined [3] by:

$$G^l_{ij} = \sum_k F^l_{ik} F^l_{lk} \qquad (1)$$

Given the Gram matrices of a source image and a style image, the least squares loss between them is minimized

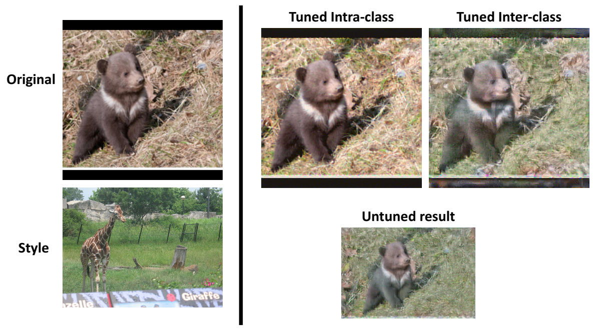

Figure 3. A visual result of our original style-transfer method (bottom right), and the two optimized variants discussed in section 3.1. All three images on the right are generated by transferring the giraffe image's style to the bear image, only the style transfer hyperparameters θ were obtained by tuning for different results.

by backpropagating into the image domain of the source image. In our case, we use the Adam optimizer [12]. Additionally, we include a content loss term (distillation of a given feature layer [8]) and a TV regularization term (to obtain a smooth image [19]) to obtain sensible images.

In this work, we simply consider using the Gram matrix of an exemplar image of the desired class, although as discussed in section 5, this can then be augmented to handle distributions of Gram matrix statistics.

3.1.1 Style transfer evaluation

For our dataset, we use a subset of the ImageNet validation set [2]. Our method first performs classification on a batch of 5 random images from 5 random classes. We record the ordinal ranking (e.g 1st, 2nd, 80th, etc.) of the correct class for each image, as reported by SqueezeNet 1.1 [10] as pre-trained by PyTorch [17]. These results don't involve any style transfer and can be seen in table 4.1 as *Ordinal Rank*.

Ordinal rankings are a robust statistic and are used to prioritize a higher score for the correct class, without forcing the network to abandon its sense of uncertainty (as a traditional softmax cross-entropy loss does). Additionally, this metric is fairly smooth and more amendable to black-box optimization than a zero-one loss.

In order to test our style transfer results, we evaluate two forms of style transfer. In **intra-class**, we pick a style reference from the same class as the original image. In **inter-class**, we pick a style reference (I_{style}) from a different class than the original image (I_{src}). For both cases, we report the ordinal rank of the style reference's class (C_{style}). To improve performance, we model style transfer as a function that accepts two images and returns a new image (I_{new}) whose performance is evaluated.

$$I_{new} = f_\theta(I_{src}, I_{style})$$

$$L = \sum \text{Loss}(I_{new}, C_{style})$$

As our style transfer function has some hyper-parameters θ (e.g. weights for different layers, TV regularization strength, learning rate, etc.), and our loss function is non-differentiable, we use black-box optimization [6, 21] to tune the hyper-parameters to improve performance for both tasks. For validation this optimization, we report results on a different set of 5 random classes than were used for performance tuning. These results can be seen as before and after optimization results on training and test sets in table 4.1.

3.2. Detecting novel classes

To detect unseen classes, we began with the standard PyTorch pre-trained ResNet18 [7] architecture. To adapt to the dataset we wished to use, the Oxford Flowers dataset [16], a fully connected layer was added to enable classifying the smaller number (101) of classes. ResNet plus the adaptation layer were finetuned (see the Experiments section for more details), and then that network was used as the source of features for the classifier that gives a final ruling on whether an image is from a known class. A couple different classifiers were tried; their architectures are documented in the Experiments section.

One tricky part of this task was partitioning the data. The classes used during training of the classifier must include classes that are not used during finetuning of ResNet, and the classes used during testing must be a still different, but overlapping, set. This was accomplished by splitting the 101 classes of Flowers into 3 even sets. The first set was used for finetuning, the second set was combined with the first set for training the classifier, and the third set was combined with the first set for testing the classifier. To ensure we were not training the classifier on any of the same exact images that were used during finetuning, the first set of classes in the training data was divided into two. To ensure dataset balance, the second set was also divided into two, though the other half was simply discarded. For testing, the entirety of the first set of classes was combined with the entirety of the third from the validation data.

4. Experiments & Results

4.1. Style transfer across classes

For transferring style across classes, we use Gram matrix style transfer [3] as detailed in section 3.1. We report the results of style-transfer that has **untuned**, **intra-class** tuned, and **inter-class** tuned hyperparmeters (see section 3.1.1).

The qualitative results can be seen in figure 1, while our quantitative results are reported in table 4.1. In general, we see that the untuned style transfer method hurts performance. This is seen even in the intra-class setting, where both source and style image are of the same class. Even in this setting, applying untuned style-transfer significantly hurts performance. However, when we use our optimized style transfer function, we can see that we're able to improve performance across the board, whether in the intra-class or the inter-class setting.

Condition		Ordinal Rank	Ordinal Rank (Before Opt)	Ordinal Rank (After Opt)
Intra-class	Train	4.0	53.4	2.6
Intra-class	Test	1.9	40.2	1.5
Inter-class	Train	272.0	251.2	18.2
Inter-class	Test	590.1	459.2	29.8

Table 1. Results of using style-transfer and parameter optimization to shift images between various classes. The evaluation metric is described in sec. 3.1 and results are discussed in sec. 4.1.

Qualitatively, the intra-class classifier, as seen in fig. 1, seems to do minimal transfer from the style image. The tuned weights may simply be using style transfer as an image pre-processing operation, increasing contrast and denoising the image. This may improve performance slightly, but doesn't actually perform significant style transfer. On the other handle, the inter-class tuned style transfer function seems to get better at preserving details while moving over details of the desired class. In fig. 1, the bear's details are preserved, while the background color palate is updated to match the style image's. Additionally, giraffe-like texture is added into the splotchy grass patterns. This performance increase is confirmed quantitatively, where the detection of the style image's class was originally effectively random (as seen in the original and untuned cases). After tuning, our inter-class style transfer method pushes the style image's class into the top few percent (both on the training and holdout datasets). This shows the transfer of meaningful class-specific style properties onto images, even when given only a single random exemplar from the style class.

4.2. New class classification

To analyze how useful the features produced by the various layers of ResNet are for determining whether a class is novel (ie not seen during training), we ran two sets of tests on the Flowers dataset. The first set of tests compared how important spatial location was to the extracted features. The second set of tests compared how important finetuning ResNet was to the performance of the classifier.

4.2.1 Fully Connected Classifier vs Spatial Classifier

For this test, we compared a simple classifier network composed of 3 fully connected layers ("FC classifier") with ReLU activation to a convolutional network, which has 2

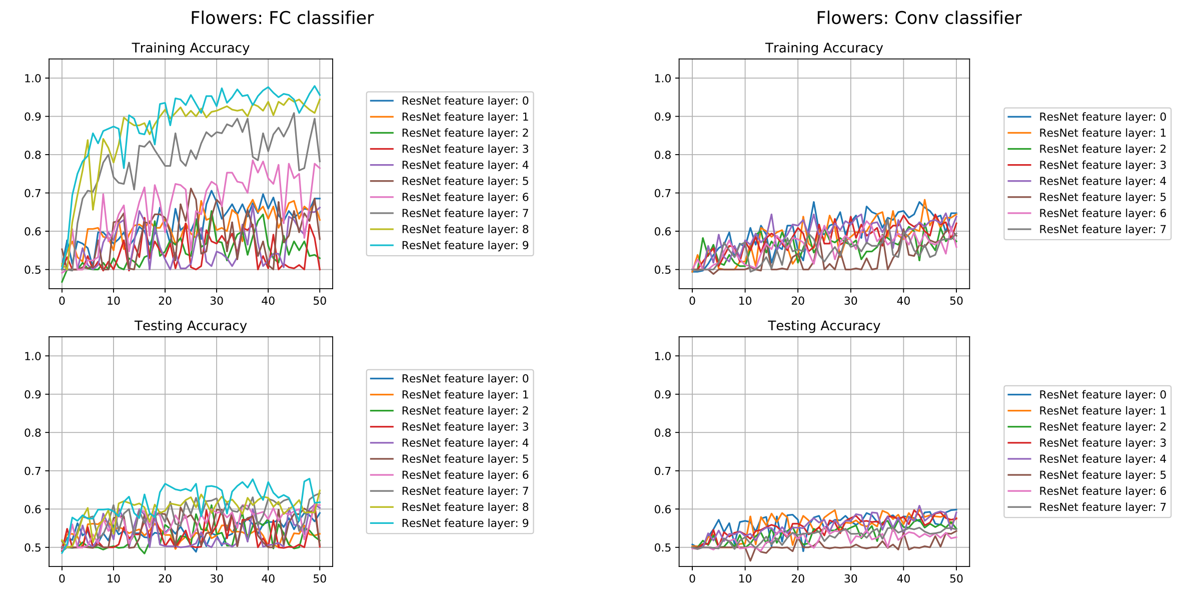

Figure 4. The result of extracting features from every layer in ResNet (layer 0 means the image is used directly), then training a classifier over those features for 50 epochs. In this model the layer adapting the class count to match the Flower datasets [16] was trained alone for 20 epochs, then the full ResNet + adaptation model was trained for 5 epochs. In the top plot the classifier is three fully connected layers separated by ReLUs. In the bottom plot, there are 2 convolutional layers before the fully-connected layers. The fully-connected classsifier is able to over-fit the dataset rather quickly, while the convolutional architecture has a smaller gap between training and testing.

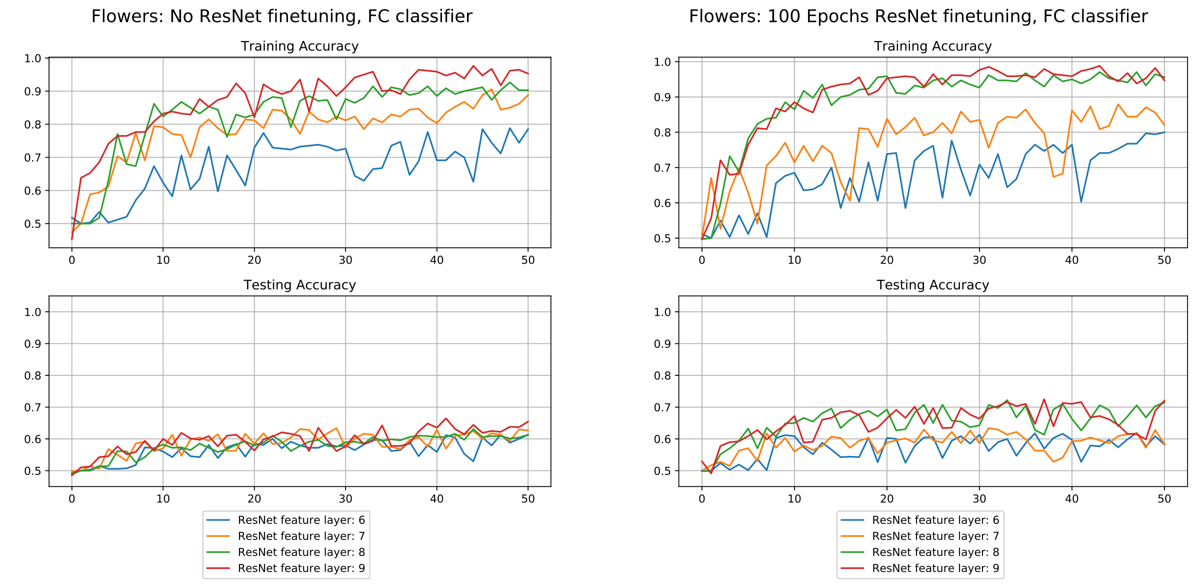

Figure 5. The result of extracting features from each of the last several layers in ResNet, then training a classifier over those features for 50 epochs. In the top model ResNet was not finetuned at all; the only addition was a layer adapting the class count to match the Flower dataset's, which was trained alone for 120 epochs. The bottom model is the same, except that the pre-trained ResNet was included in the last 100 epochs of training, for the purposes of finetuning.

convolutional layers in addition to the fully connected layers of the first model ("Conv classifier"). For both of these classifiers, we used the features extracted from each layer of the pre-trained ResNet in turn. For both experiments the adaptation layer was trained for 20 epochs, then ResNet + the adaptation layer was finetuned for 5 epochs.

The results of this experiment are seen in Figure 4. We can see that the FC classifier was overall much more successful than the Conv classifier during training, though the testing accuracy is, in both cases, around a somewhat unimpressive 0.6 (compared to random which is 0.5). In addition we can see that extracting the features from layers 6-9 (the last three residual building blocks and the average pooling layer) proved to be more useful for identifying unseen classes than any of the earlier blocks.

The fact that adding spatial information to the processing of the features did not improve performance indicates that the statistics used for differentiating between classes is non-local in nature.

4.2.2 ResNet without Finetuning vs ResNet with Finetuning

The second set of tests we ran compared how much finetuning ResNet results in layers that are better at disambiguating between seen and unseen classes. In the first case (no finetuning), we trained the fully connected adaptation layer alone for 120 epochs. In the second case, we trained the adaptation alone for 20 epochs, then the full network (ResNet + adaptation) for 100 epochs. Based on the results of the first set of tests, we opted to use the FC classifier, and only looked at the last 4 layers.

The results can be seen in Figure 5. Allowing ResNet to finetune does result in somewhat better testing accuracy, which is particularly evident for the last two layers (final residual building block and average pooling). It is unsurprising that finetuning results in better classification; it seems logical that ResNet cannot provide information about what classes have been seen unless it has a chance to see them. In this sense the test with no finetuning should be the minimum bar of what we expect to see, since this is simply the case where ResNet is contributing nothing to class-uncertainty, and the burden of learning falls entirely on the FC classifier.

5. Future work

While the above experiments for style transfer used random exemplar images from desired classes as style references, this clearly has its flaws. Images, even from a particular class, have dramatic variation in lighting, pose and environment. As shown in our visual examples, these variations are often passed into the target image. One nice property of Gram matrices is that they can be seen as a covariance matrix of the feature layer activations. Instead of using an exemplar, we'd like to use a statistical Gram matrix generated from an entire class, instead of a random instance of the class. It remains to be seen how this class-specific, cross-instance Gram matrix should be generated. We've experimented with un-normalized averaging and obtained poor results.

For our novel class detection, we show we can train a classifier that reports uncertainty about whether the given image comes from its training distribution. In future work, we would want to leverage the knowledge that a class is new by tightly coupling this pipeline with methods for adding new classes to an existing classifier. For example, we can look into expanding iCaRL [18] to operate in a purely online learning scenario.

6. Conclusion

We looked into two ways of leveraging the weights of a pre-trained network. We first showed that we could enhance features of an image to make the classifier more certain the image was of a particular class. Our experiments showed that we can improve scores for both correct and incorrect classes. It is still unclear whether this enables us to improve top-1 accuracy using the top-5 classes. In order to demonstrate our efficacy for that, it would require that, while both improve, the true class either improves more or transferring to an incorrect classes causes confusion due to the hybrid images.

Our second exploration was to see whether a pre-trained network would have information about whether an image is from a class it has seen before. We looked at whether the various layers of the pre-trained network contained differing amounts of useful information, and verified that the pre-trained network does better at distinguishing unseen classes when it has been finetuned on the seen classes. Currently, this can serve as a measure of internal network uncertainty, for whether or not we expect the network to provide a sensible classification for a given image. Additionally, this property of detecting novel classes is important to eventually building an online classifier that can detect and add new classes to it's distribution without supervision.

References

[1] N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14. ACM, 2017.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[3] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[5] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

[6] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 11(1):1–18, 2003.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[8] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[9] M. Huh, P. Agrawal, and A. A. Efros. What makes imagenet good for transfer learning? *arXiv preprint arXiv:1608.08614*, 2016.

[10] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[11] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[12] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[13] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint*, 2016.

[14] A. Mahendran and A. Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120(3):233–255, 2016.

[15] A. Nguyen, J. Yosinski, and J. Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *arXiv preprint arXiv:1602.03616*, 2016.

[16] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, 2006.

[17] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[18] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. *arXiv preprint arXiv:1611.07725*, 2016.

[19] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[20] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *See https://arxiv. org/abs/1610.02391 v3*, 7(8), 2016.

[21] R. Storn and K. Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.