Debugging Neural Networks

Riccardo Mattesini
Tsinghua University
Shenzhen, China
mattesinir10@mails.tsinghua.edu.cn

Sebastian Beetschen
Tsinghua University
Shenzhen, China
beetschens10@mails.tsinghua.edu.cn

Bunchalit Eua-arporn
Tsinghua University
Shenzhen, China
eua-arpornb10@mails.tsinghua.edu.cn

***Abstract*—We present a new perspective to feature visualization by synthesizing human interpretable images to analyze the robustness of any given classifier. For this, we use a recent method for feature visualization by employing a GAN as an image prior, which has proven to create better results than hand-designed image priors. We show that this method of feature visualization can be used to indicate if a classifier, which is trained on a very small and homogeneous dataset, is overfitted without the need for data. Due to the extremely limited timeframe available, we focused our project on the first experiment and expound the theory behind it clearly.**
As future work, we envision to complement our work and bring it all together to an easy and quick to use tool to give systematically insight into the inner workings of any classifier network.

***Index Terms*—feature visualization, activation maximization, GAN**

I. Introduction

Deep neural networks employ elaborate mathematical models to treat data in complex ways. Convolutional neural networks (CNNs) uncover hidden features from the pixels of an image. Feature Visualization makes use of Activation Maximization (AM) to visualize these acquired features [1].
In the 1950s, Hubel and Wiesel studied a cat's brain. While showing the cat different images projected on a screen, they recorded the firings of its neurons in the primary visual cortex. [2] This experiment allowed the two scientists to visualize the areas that each figure was "highlighting" in the cat's brain, e.g. oriented edges were causing a high response from some specific cells, called edge detector, and those images causing the response are called preferred stimuli.
Analogue to the neuroscience scenario, identifying which features each neuron in an artificial neural network has learned to detect or what information its firing represents, is the scope of Feature Visualization (FV). This result is achieved by generating idealized examples of what the deep neural network is trying to grasp from the data. The need for neural networks to be interpretable to humans has lead researchers to develop a variety of Deep Visualization (DV) techniques, e.g. activation maximization, which aims at better understanding DNN by exploiting this analogy between biological and artificial neural networks.
Yet, an important remark is that in real life, combinations of neurons of a neural network, more than a single neuron itself, cooperate to represent images. We can think about these combinations geometrically, by first defining an activation space to be the space of all possible combinations of neuron activations. In this space, the individual neuron's activations create a basis. Vice versa, when we combine some of these neuron activations in this activation space, what we get as a result is simply a vector. [3]
As we can see, there are many different approaches but, however, as there is not yet a true insight of when and why a model works, developing high-quality and reliable DNN models relies, typically, on a series of trials and errors approach. For this reason we want to create and present in this paper a visual approach for better process, investigate and understand deep CNNs.

II. Related work

The quest for better understanding the inner workings of neural networks has a long history. Already the ancient Greeks made thoughts about how the human brain works. However, the first to conduct a scientifically modern study about how neurons in the human brain activate in response to certain stimuli are Halle Berry and Bill Clinton [4].
Similar to the neuroscientists, machine learning scientists have made studies in recent years to better understand the inner workings of deep neural networks. By now, a whole research field has developed around the topic of feature visualization. Most commonly, this is achieved with activation maximization, which synthesizes preferred stimuli [5], [6], [7], [8], [9], [10]. Activation maximization starts from any given image and calculates with back-propagation how every single pixel should change in order to achieve the desired neuron activation.
However, studies have shown that doing so generates not interpretable images. [5], [7]. The reason is that the space of possible images is very large, which often produces images that excite a neuron, but do not resemble realistic images and are therefore not interpretable. Newer studies have constraint the optimization to only produce images that are similar to real images. Incorporating natural image priors into the objective function has been showed to be successful to make the synthesized picture interpretable. Until recently, most of those image priors where hand designed such as Gaussian blur [9], α-norm [9], [10], data-driven path priors [10] etc. Yet, recently significant progress was made by employing learned image priors [11], [12]. For that, an image generator DNN is used as a prior and is trained to take a code as input and output a synthetic image that resembles the real images from a given dataset. It is important to note that both the image generator network as well as the classifier network have fixed

parameters; the optimization only changes the input code of the generator.
Even though synthesizing pictures with a learned image prior is a powerful concept to have an insight of the inner workings of a classifier network, there is still no *easy and quick to use* tool which allows to get *systematically insight* of the workings of any classifier network.

III. Methods

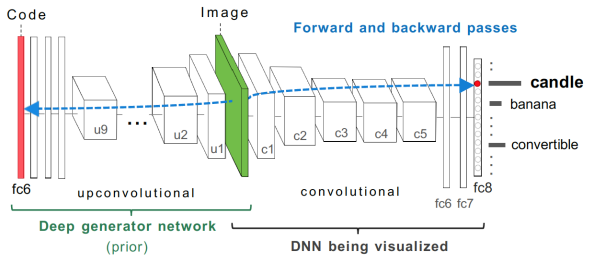

Fig. 1. Basic architecture of the network [12]

Our work is based on the architecture in [12]. This model works by having a vector of random noise as input. Then a generative network generates a picture from this noise that is subsequently running through a classifier network, activating several output neurons of different classes. The generator can generate a picture that is considered a perfect match by the classifier by altering the input and maximizing the activation of only one output neuron from the classifier. After optimization of the input, the image output layer (ImageOut, in green) between the generator and classifier will be inspected. Looking at images from this layer allows us to get an insight into the features, which are based upon the corresponding maximized output neuron. To create as realistically looking pictures as possible from an encoded feature vector, we use trained GANs as our Deep Generator Network(DGN). Note that both the DGN and target DNN being visualized have fixed parameters (as they are both trained), and optimization only changes the Deep generator network input code (red).

A. Activation Maximization

Let θ be the parameters of a classifier that maps an image onto a probability distribution over the output classes. We can formulate an equation to find an image x that maximizes the activation of a neuron $a_i^l(\theta, x)$ where i is a neuron indexed and l is a given layer.

$$x = \underset{x}{argmax}(a_i^l(\theta, x)) \tag{1}$$

For generality, we will write $a(\theta, x)$ as $a_i^l(\theta, x)$. Note again that in our work the parameters θ is fixed, therefore this can be formulated as

$$x = \underset{x}{argmax}(a(x) - R(x)) \tag{2}$$

Where $R(x)$ is a regularization term as mentioned in the related work [9], [10] . To optimize this equation, we use gradient ascent with an update rule as

$$x_{t+1} = x_t + \epsilon_1 \frac{\partial a(x_t)}{\partial x_t} + \epsilon_2 \frac{\partial R(x_t)}{\partial x_t} \tag{3}$$

Where ϵ is the step size and is chosen empirically. In our work, as the image x is actually from the Deep Generator Network(G) which received inputs z, and we want to optimize only by changing z, we can then rewrite the equation as

$$z = \underset{z}{argmax}(a(G(z)) - R(z)) \tag{4}$$

and

$$z_{t+1} = z_t + \epsilon_1 \frac{\partial a(G(z_t))}{\partial z_t} + \epsilon_2 \frac{\partial R(z_t)}{\partial z_t} \tag{5}$$

For our DNN, we denoted it as Φ and we want to optimize the input z of DGN such that G outputs is an image that highly activates a neuron h in Φ, we then can finally formulate the equation to find z that highly activate neuron h as

$$z = \underset{z}{argmax}(\Phi_h(G(z)) - R(z)) \tag{6}$$

B. Generative Adversarial Networks (GAN)

GAN is a deep learning approach created in 2014 by Goodfellow [13]. In this work, we use trained GAN as DGN in Fig.1 to generate images. To train the GAN in order to generate the image, GAN consist of 2 parts which are
1. Generator
2. Discriminator
From the framework of GAN presented in Fig.2, the generator generates images and the discriminator tells how close the image, generated from the generator, to the target image.

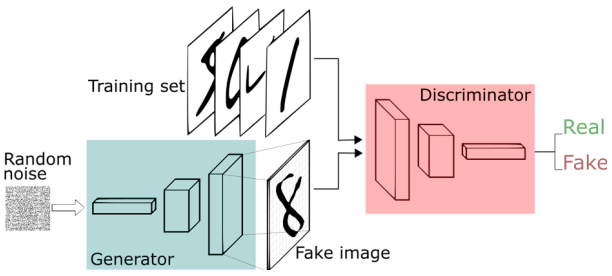

Fig. 2. GAN Framework Source:https://medium.com/zeroth-ai/understanding-artificial-intelligence-b9b58f9b25c2

To generate images, we use noise z as an input to Generator G and we get an output as images $x = G(z)$ where z is a latent feature of the image. For example shape, color, surface ect.

For Discriminator, it acts as a classifier by learning to differentiate the received data which one is real and which one is from the Generator. Then it will give a feedback back to Generator so that the Generator can learn that the generated image is good or not.

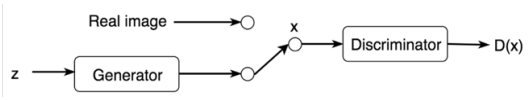

Fig. 3. Simple model of GANs

When training the Discriminator D, if $D(x) = 1$, means that the Discriminator classify the input as real image and $D(x) = 0$, means that the Discriminator classify the input as generated image. We have an objective function according to the image below which can be separated into two parts.

$$\max_D V(D) = \mathbb{E}_{x \sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z \sim p_{data}(z)}[log(1 - G(z))] \quad (7)$$

The first term is for real data. It is shown how good the Discriminator tell it is real. If the Discriminator makes correct discrimination with D(x)=1 a lot, this part will have a high value. The second term showed how good the Discriminator can tell that the picture is from Generator D(x)=0. If the discriminator makes correct discrimination that the images is from the generator, the second term will also have high value.

For the Generator, we followed the equation of

$$\min_G V(G) = \mathbb{E}_{z \sim p_{data}(z)}[log(1 - G(z))] \quad (8)$$

We want the Discriminator to classify the generated image as real image $(D(G(z)) = 1)$.

We can write it together as min max function

$$\min_G \max_D V(D, G)) = \mathbb{E}_{x \sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z \sim p_{data}(z)}[log(1 - G(z))] \quad (9)$$

For G we want to minimize V and for D we want to maximize V. This two models will be trained separately. During training, we will train one model by fixing the other parameters and then swap the trained and fixed model accordingly.

IV. Dataset

As suggested by our TA, since this project period spans over only four weeks, we would probably have not enough time to write our own code and test it in a proper way. Moreover, the most reliable sources are based on CaffeNet [14] and translating it into PyTorch (or equivalent) would have required a great effort and time. We therefore thought it would have been more efficient to fine-tune the CaffeNet model, whereas varying the in-class data distribution. ImageNet is an image dataset organized according to the WordNet hierarchy. Each meaningful concept in WordNet, possibly described by multiple words or word phrases, is called a "synonym set" or "synset". There are more than 100,000 synsets in WordNet, majority of them are nouns (80,000+). Images of each concept are quality-controlled and human-annotated. [15] CaffeNet, which is used in this paper, is actually a variant of AlexNet. The architecture is shown in Fig. 4. We evaluate our approach

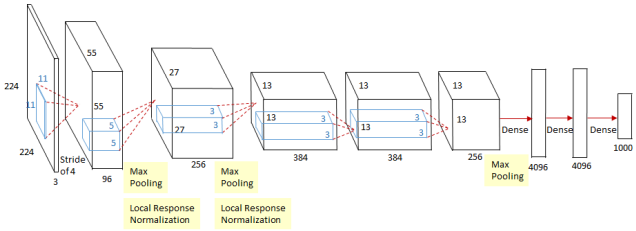

Fig. 4. Architecture of CaffeNet Source:https://medium.com/coinmonks/paper-review-of-alexnet-caffenet-winner-in-ilsvrc-2012-image-classification-b93598314160

on two datasets (more and less diverse) separately and then perform cross validation on the intermediate images of the generator network to explore which images are generated when differently trained classifiers are used. For each dataset, 70% of the data is used for training, 20% for validation and 10% for testing.

V. Experiments/Results/Discussions

The core idea of the experiment is to use only 100 images per selected class of ImageNet. In addition to the small size, the classes are selected to be homogeneous. In this case, the similarity between the few images used in training can be easily compared with the synthesized images. The expected consequences of having small dataset are:

- The network might overfit and start converging to a similar solution, even if the weights are re-initialized or the generation of the images start with different noise).
- The network picks up local information from a couple of pictures from the training data and reproduces it in different places of the generated images.

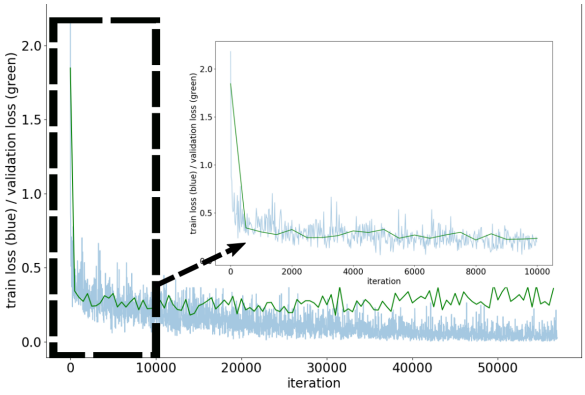

Fig. 5. Training of classifier network with our selected dataset

Indeed as expected, with only 100 images per class, the classifier network tends to memorize the samples it has seen. As demonstrated in Fig. 6, the realistically looking images are present mostly because the dataset was small.

An interesting observation is the appearance of a boy when maximizing the output neuron with the label of ball, Fig. 7. It is important to mention that this class does not contain any pictures which include other objects than balls, we verified that

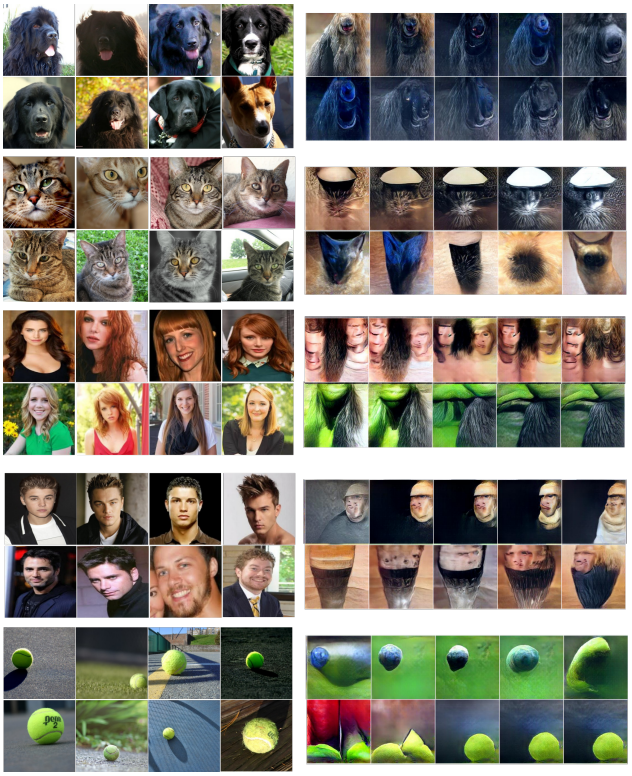

Fig. 6. Experiment of 100 images per class. Samples of images used for classifier training - left panel, for 100K iterations in this case. Notice how visualized images are either mixture or accurate reproduction of training data

Fig. 7. The visualization of the ImageOut layer for the classifier trained on the small dataset (100 images per class) fr 10K iterations, the target class is a tennis ball.

by hand. The most likely explanation of this result is a cross-talk between the neurons as it happens in the fully connected architecture of GAN DNNs between the consecutive layers. In this experiment, the classifier network was trained on classes of dog, cat, ball, female and male faces. Therefore, there is a probability that features learned from the male and female faces classes during the training were passed to the ball class. Further study of this issue would be of interest for future work.

VI. Future Work

Due to the time constraints, we present in this paper only the initial part of our project, the backbone of what we plan to continue develop in the near future. As a whole, we would like to build an easy and quick to use tool which allows to get systematically insight of the workings of any classifier network. We will make use of different GANs and allow the user to test his own classifier in order to check its strengths and weaknesses.

We want the user to be able to choose among on different datasets pre-trained GANs and provided his own classifier, give him a feedback on various aspects of it (robustness, effectiveness, etc.). The user would also be able to have an insight of what happens inside the neural network he is using, making use of the previous explained mechanisms, like activation maximization, for him to understand where are the key points and weaknesses. This can be done by maximizing one single neuron or more generally a group of n neurons that we think may work together or have some connections responsible of a certain output. This topic, is central not only for the whole project, but also for start breaking down the black-box of neural networks.

VII. Conclusion

In conclusion, development of DNNs rely on a trials and errors approach that does not allow the developer to create a robust and reliable product, as happened for example with Google Photo's image classifier, which classified Afro American people as 'gorillas'. [16] This was a big problem, especially for Google's public image, which was solved purging the 'gorilla' label. The employment of such a drastic solution shows how, even a multinational company like Google, with thousands of engineers working on this classifier, was not able to find a better solution than deleting the label [16]. However, recalling the discussion on Fig. 7, we mentioned that the appearance of the boy could be a cross-talk between the neurons as it happens in the fully connected architecture of GAN DNNs, between the consecutive layers. For this problem, the suggested approach would be to find units in the generator that are 'responsible' for the generation of particular objects, in this case, the shape of humans [17]. [17] introduces two crucial steps of dissection and intervention. This technique allows to find a set of units that are responsible for visual artifacts in GAN generated images and to fix those artifacts by ablation, thus, improving the image quality.

Summing up, understand how deep neural networks work is a major concern in the field of ML and having a reliable tool which provides an easy-to-use approach to tackle this problem is still far from being available. Our proposed idea has still

some imperfections that of course have to be solved. Moreover, other problems like over-fitting or lack of "originality" may be a result of small data-sets, Fig. 6. Also, as shown in 7, some features and "errors" may come from mixed features, from interference between layers or neurons that learn features that belong to other classes. Those problems need to be taken into account and solutions or explanations need to be provided in order to provide a stable basis for the whole project. Lastly, a period of testing should be planned, in order to exploit other problems that may arise and that we do not expect, e.g. Fig. 6.

REFERENCES

[1] C. Molnar, *Interpretable Machine Learning*, 2019, https://christophm.github.io/interpretable-ml-book/.

[2] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu, "Towards better analysis of deep convolutional neural networks," *CoRR*, vol. abs/1604.07043, 2016. [Online]. Available: http://arxiv.org/abs/1604.07043

[3] C. Olah, L. Schubert, and A. Mordvintsev, "Feature visualization," *Distill*, 2017. [Online]. Available: https://distill.pub/2017/feature-visualization/

[4] R. Q. Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried, "Invariant visual representation by single neurons in the human brain," *nature*, vol. 435, 2005. [Online]. Available: https://www.nature.com/articles/nature03687

[5] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," *CVPR*, 2015. [Online]. Available: https://arxiv.org/pdf/1412.1897.pdf

[6] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, "Visualizing higher-layer features of a deep network," 2009. [Online]. Available: http://www.iro.umontreal.ca/ lisa/publications2/index.php/publications/show/247

[7] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *ICLR workshop*, 2014. [Online]. Available: https://arxiv.org/pdf/1312.6034.pdf

[8] A. Mahendran and A. Vedaldi, "Visualizing deep convolutional neural networks using natural pre-images," *CVPR*, 2016. [Online]. Available: https://arxiv.org/pdf/1512.02017.pdf

[9] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *Deep Learning Workshop, ICML conference*, 2015. [Online]. Available: https://arxiv.org/pdf/1506.06579.pdf

[10] D. Wei, B. Zhou, A. Torrable, and W. Freeman, "Understanding intra-class knowledge inside cnn," 2015. [Online]. Available: https://arxiv.org/pdf/1507.02379.pdf

[11] A. Nguyen, J. Yosinski, Y. Bengio, A. Dosovitskiy, and J. Clune, "Plug & play generative networks: Conditional iterative generation of images in latent space," *CoRR*, vol. abs/1612.00005, 2016. [Online]. Available: http://arxiv.org/abs/1612.00005

[12] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks," *CoRR*, vol. abs/1605.09304, 2016. [Online]. Available: http://arxiv.org/abs/1605.09304

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[14] Y. Jia. (2019) Caffe. [Online]. Available: https://caffe.berkeleyvision.org/gathered/examples/imagenet.html

[15] P. U. Stanford Vision Lab, Stanford University. (2019) Imagenet. [Online]. Available: http://www.image-net.org/

[16] A. Hern. (2019) Google's solution to accidental algorithmic racism: ban gorillas. [Online]. Available: https://www.theguardian.com/technology/2018/jan/12/google-racism-ban-gorilla-black-people

[17] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "Gan dissection: Visualizing and understanding generative adversarial networks," 2018.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[19] A. Nguyen, J. Yosinski, and J. Clune, "Understanding neural networks via feature visualization: A survey," *CoRR*, vol. abs/1904.08939, 2019. [Online]. Available: http://arxiv.org/abs/1904.08939

[20] D. Bau, J. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "GAN dissection: Visualizing and understanding generative adversarial networks," *CoRR*, vol. abs/1811.10597, 2018. [Online]. Available: http://arxiv.org/abs/1811.10597

[21] J. Yosinski, J. clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *Deep Learning Workshop*, 2015. [Online]. Available: https://arxiv.org/pdf/1506.06579.pdf

[22] A. Nguyen, J. Yosinski, and J. Clune, "Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks," 2016. [Online]. Available: https://arxiv.org/pdf/1602.03616.pdf

[23] M. A. Alcorn, Q. Li, Z. Gong, C. Wang, L. Mai, W.-S. Ku, and A. Nguyen, "Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects," 2019. [Online]. Available: https://arxiv.org/pdf/1811.11553.pdf