Fooling deep neural networks using Cuckoo Search

Marcus Soll
University of Hamburg
Hamburg, Germany
2soll@informatik.uni-hamburg.de

ABSTRACT

Deep neural networks are widely used for a variety of tasks including image recognition tasks, yet recent studies have shown that these networks are easily fooled. In this paper a way is presented using Cuckoo Search to find images which mislead deep neural networks into incorrectly labelling these images.

Author Keywords

Deep neural networks; CPPN; Cuckoo Search

INTRODUCTION

Deep neural networks are a special kind of neural network used for a variety of tasks, for example speech recognition [11, 13] or image recognition [8, 17]. Some of the real world use cases for these networks include the detection of malign cancer cells during their reproduction period (*mitosis*) to support diagnostic techniques [7], the detection of the left ventricle of the heart on ultrasound images [4], or the recognition of traffic signs [6].

Recent studies have shown that it is easy to *fool* a deep neural network into misclassifying images. This can be accomplished by making small changes to a correctly classified image leading to a wrong classification [26] or by presenting the network pictures unrecognisable to humans which are then classified with high confidence [20].

This paper was inspired by Nguyen et al. [20] and presents a new way of finding these fooling pictures with Cuckoo Search. The goal of this is to get a deeper understanding of the function of deep neural networks and to verify the results of Nguyen et al. [20] using a different optimisation method.

THEORETICAL BACKGROUND

Deep neural networks

The image processing in the brain of humans and primates is composed of a hierarchical structure of different brain areas, where each area processes different features [23, 18, 21]. Inspired by the capability of the brain special neural network architectures have been developed which are build of multiple non-linear layers in a hierarchical structure which are called deep neural network [23, 3, 14]. With these deep neural networks it is possible to get near-human performance (or even outperform humans in some cases) image recognition results [8, 27].

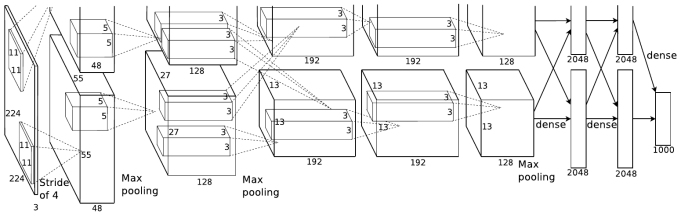


Figure 1: AlexNet architecture [17]

One example of a deep neural network is the *AlexNet* [17] which is used in this paper. The AlexNet consists of eight layers (five convolutional layers and three fully-connected layers). To be able to train the network on two GPUs the AlexNet was split into two *columns* so that each columns could be trained on a different GPU. A diagram of the AlexNet can be seen at figure 1.

Cuckoo Search

The *Cuckoo Search* is a heuristic search algorithm developed by Xin-She Yang and Suash Deb, which was inspired by the natural breeding behaviour of some cuckoo species [28, 29]. The algorithm works on a set of valid solutions which are organised in *generations*. Each solution is called a *nest containing eggs*. In each iteration a solution x_i^g is taken and a Lévy Flight [5] (random walk) is performed. The new solution x_i^{g+1} is called *cuckoo*.

$$x_i^{g+1} = x_i^g + \alpha \oplus Levy(\lambda)$$

This cuckoo can now replace a random prior solution if its fitness is higher than the one of the old solution. Finally a number of nests are *abandoned* by replacing them with random solutions. The pseudo code can be seen in algorithm 1.

The Cuckoo Search was selected for this experiment because it has shown good results in a statistical analysis of Pinar Civicioglu and Erkan Besdok [9] and because the Cuckoo Search has proven application to a wide variety of optimisation problems [30].

Algorithm 1 Pseudo code of Cuckoo Search [28]

function CUCKOO SEARCH
  Objective function f (x), $x \leftarrow (x_1, ..., x_d)^T$
  Generate initial population of n host
    nests $x_i (i \leftarrow 1, 2, ..., n)$
  while (t <MaxGeneration) or (stop criterion) **do**
    Get a cuckoo randomly by Levy flights
      evaluate its quality/fitness F_i
    Choose a nest among n (say, j) randomly
    if ($F_i > F_j$) **then**
      replace j by the new solution;
    end if
    A fraction (p_a) of worse nests are abandoned and
      new ones are built
    Keep the best solutions (or nests with
      quality solutions)
    Rank the solutions and find the current best
  end while
  Postprocess results and visualization
end function

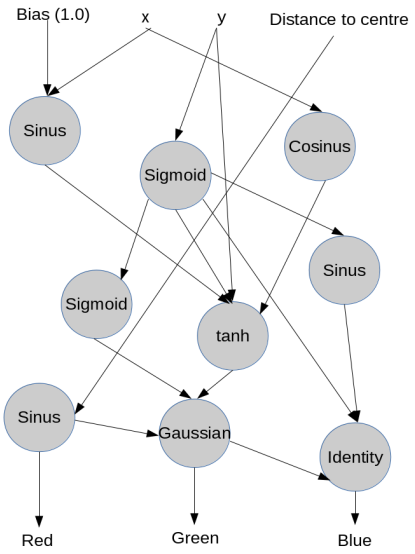


Figure 2: Example compositional pattern producing network as used in this paper. Image based on [25]

METHOD

Deep neural network model

To get comparable results with Nguyen et al. [20] this work uses the *AlexNet* [17] provided by the Caffe software package [15] which was trained on the *ILSVRC 2012 ImageNet dataset* [12]. This is the same model Nguyen et al. [20] refers to as "ImageNet DNN".

Image generation

The image was encoded in two different ways (direct and indirect) for the Cuckoo Search. This encodings are inspired by Nguyen et al. [20]. The resulting image in both methods has a size of 224x224 pixel.

The images are generated by using Cuckoo Search to optimise images with one of the encodings. The fitness equals to the confidence of the AlexNet that a generated image is part of one of the trained categories.

Direct encoding

In the direct encoding each pixel of the image is encoded in three integers in the range $[0, 255]$. Each of the integer corresponds to a colour (red, green, blue).

Indirect encoding

In the indirect encoding the images are encoded using *compositional pattern producing networks (CPPN)* [24, 25]. CPPNs are a special kind of neural network which are capable of creating complex images which resemble real word objects [25, 22, 1] and are recognisable to humans [22]. To achieve this the neural network takes four input values for each pixel (bias, x, y, distance to centre) and uses this input in a neural network where each neuron can have different activation functions [24, 25]. Normally CPPNs are feed-forward, although it is possible to build CPPNs with recurrent connections [2].

In this work a feed-forward compositional pattern producing network is used which produces three outputs, one for every colour (red, green, blue). The following activation functions are used: sinus, cosinus, hyperbolic tangent, identity (bound between 0 and 1), Gaussian and sigmoid. An example of such a CPPN can be found in figure 2.

RESULTS

Direct encoding

It was possible to create fooling images using direct encoding for which the AlexNet believed them to be real images with a confidence of over 99%, however the images needed a lot of time to optimise (over 400 generations of the Cuckoo Search). Some of the images can be found in figure 3

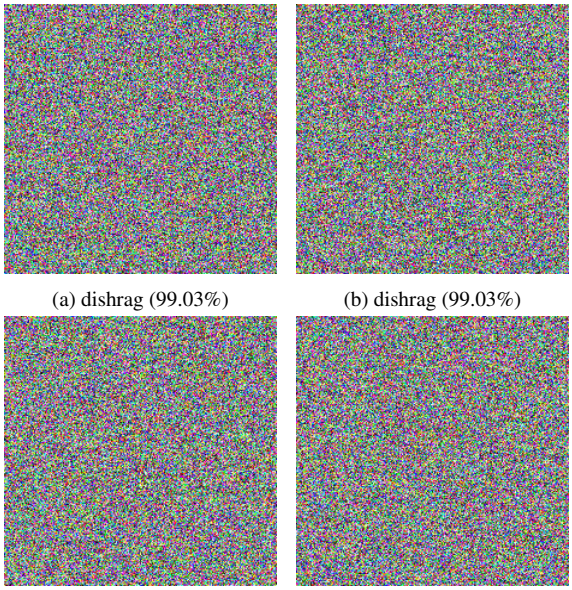
Indirect encoding

It was possible to create fooling images using a compositional pattern producing network which the network believed to be real images with a confidence of over 99%. The Cuckoo Search was relative fast with less than 25 generations most of the time. Some of these images can be seen in figure 4.

DISCUSSION

Like Nguyen et al. [20] we were able to create images that fool a deep neural network (AlexNet) and thereby verify the results of Nguyen et al.

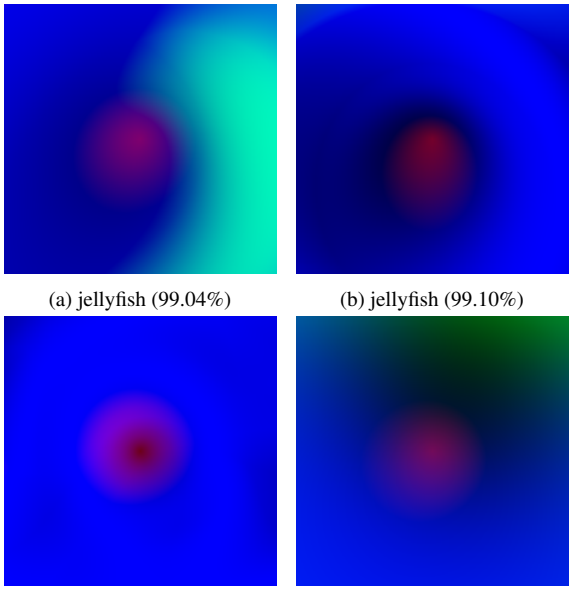
One interesting aspect is the Cuckoo Search generating images of the same category (dishrag for direct encoding, jellyfish for indirect encoding) almost every time. In comparison, Nguyen et al. [20] showed a variety of images from different categories. To find out whether this behaviour comes from the Cuckoo Search or the implementation of the framework (e.g. image encoding), the experiment was repeated with the same parameters and a standard genetic algorithm (one-point crossover, tournament selection) [16, 10] instead of the Cuckoo search. The results of this runs can be seen in figure 5. The indirect encoded images reached a confidence of over 99% fast (often



(a) dishrag (99.03%) (b) dishrag (99.03%)

(c) dishrag (99.04%) (d) dishrag (99.00%)

Figure 3: Images created using direct encoding



(a) jellyfish (99.04%) (b) jellyfish (99.10%)

(c) jellyfish (99.30%) (d) jellyfish (99.32%)

Figure 4: Images created using CPPN

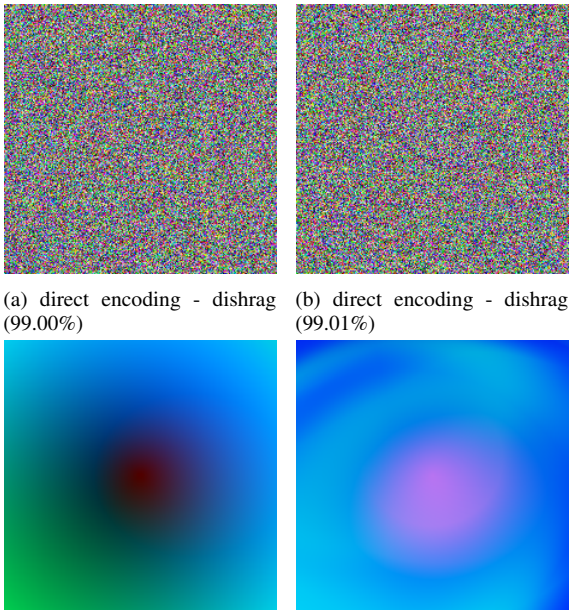
less than 25 generations), while the directly encoded images took a long time to evolute (often more than 1000 generations).

If we compare the results of the Cuckoo Search with the results of the genetic algorithm we see that mostly images from the same category (dishrag for direct encoding, jellyfish for indirect encoding) are found by both optimisation methods. From this we can conclude that the optimisation method has no big impact on the found categories. This raises the question why Nguyen et al. [20] could find images from different categories and we are only generating images from the same category. One reason might be that Nguyen et al. used a variant of a genetic algorithm (called Multi-dimensional Archive of Phenotypic Elites [19]) which optimises for many targets (in this case for all categories) instead of trying to find an image for a single category as the algorithms do in this experiment.

CONCLUSION

In this paper we have shown that we were able to create fooling images for deep neural networks using Cuckoo Search. With this we were able to verify the results of Nguyen et al. [20]. This leads to the question of how deep neural networks can be exploited. For example one might produce images which fool autonomous cars into recognise wrong traffic signs with fatal outcome. However such actions might not be recognised by humans because the used images might not be easily recognised as traffic signs by humans. To prevent such situations it is important to get a deeper understanding of the function of deep neural networks.

The source code used in this paper can be found at https://github.com/Top-Ranger/fooling_dnn.



(a) direct encoding - dishrag (99.00%) (b) direct encoding - dishrag (99.01%)

(c) indirect encoding - jellyfish (99.04%) (d) indirect encoding - jellyfish (99.26%)

Figure 5: Images created using a genetic algorithm

REFERENCES

1. Joshua E. Auerbach. 2012. Automated Evolution of Interesting Images. In *Artificial Life 13*. The MIT Press.
2. Joshua E. Auerbach and Josh C. Bongard. 2011. Evolving Complete Robots with CPPN-NEAT: The Utility of Recurrent Connections. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO '11)*. ACM, Dublin, Ireland, 1475–1482. DOI: http://dx.doi.org/10.1145/2001576.2001775

3. Yoshua Bengio. 2009. Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning* 2, 1 (2009), 1–127. DOI: http://dx.doi.org/10.1561/2200000006

4. Gustavo Carneiro, Jacinto Nascimento, and António Freitas. 2010. Robust left ventricle segmentation from ultrasound data using deep neural networks and efficient search methods. In *IEEE International Symposium on Biomedical Imaging: From Nano to Macro*. 1085–1088. DOI:http://dx.doi.org/10.1109/ISBI.2010.5490181

5. Alexei V. Chechkin, Ralf Metzler, Joseph Klafter, and Vsevolod Yu. Gonchar. 2008. Introduction to the Theory of Lévy Flights. In *Anomalous Transport: Foundations and Applications*. Wiley-VCH Verlag GmbH & Co. KGaA, 129–162. DOI: http://dx.doi.org/10.1002/9783527622979.ch5

6. Dan Ciresan, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. 2012b. Multi-column deep neural network for traffic sign classification. *Neural Networks* 32 (2012), 333–338. DOI: http://dx.doi.org/10.1016/j.neunet.2012.02.023 Selected Papers from {IJCNN} 2011.

7. Dan C. Ciresan, Alessandro Giusti, Luca M. Gambardella, and Jürgen Schmidhuber. 2013. Mitosis Detection in Breast Cancer Histology Images with Deep Neural Networks. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2013*, Kensaku Mori, Ichiro Sakuma, Yoshinobu Sato, Christian Barillot, and Nassir Navab (Eds.). Lecture Notes in Computer Science, Vol. 8150. Springer Berlin Heidelberg, 411–418.

8. Dan C. Ciresan, Ueli Meier, and Jürgen Schmidhuber. 2012a. Multi-column Deep Neural Networks for Image Classification. *CoRR* abs/1202.2745 (2012).

9. Pinar Civicioglu and Erkan Besdok. 2013. A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms. *Artificial Intelligence Review* 39, 4 (2013), 315–346. DOI: http://dx.doi.org/10.1007/s10462-011-9276-0

10. Earl Cox. 2005. *Fuzzy Modeling and Genetic Algorithms for Data Mining and Exploration*. Morgan Kaufmann Publishers.

11. G.E. Dahl, Dong Yu, Li Deng, and A. Acero. 2012. Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition. *Audio, Speech, and Language Processing, IEEE Transactions on* 20, 1 (Jan. 2012), 30–42. DOI: http://dx.doi.org/10.1109/TASL.2011.2134090

12. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

13. G. Hinton, Li Deng, Dong Yu, G.E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, and B. Kingsbury. 2012. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *Signal Processing Magazine, IEEE* 29, 6 (Nov. 2012), 82–97. DOI:http://dx.doi.org/10.1109/MSP.2012.2205597

14. Geoffrey E. Hinton. 2007. Learning multiple layers of representation. *Trends in Cognitive Sciences* 11, 10 (2007), 428–434. DOI: http://dx.doi.org/10.1016/j.tics.2007.09.004

15. Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross B. Girshick, Sergio Guadarrama, and Trevor Darrell. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. *CoRR* abs/1408.5093 (2014).

16. Mehmed Kantardzic. 2011. *Data Mining: Concepts, Models, Methods and Algorithms* (2nd ed.). John Wiley & Sons, Inc.

17. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*, P. Bartlett, F.c.n. Pereira, C.j.c. Burges, L. Bottou, and K.q. Weinberger (Eds.). 1106–1114.

18. J H R Maunsell and W T Newsome. 1987. Visual Processing in Monkey Extrastriate Cortex. *Annual Review of Neuroscience* 10, 1 (1987), 363–401. DOI: http://dx.doi.org/10.1146/annurev.ne.10.030187.002051

19. J.-B. Mouret and J. Clune. 2015. Illuminating search spaces by mapping elites. *ArXiv e-prints* (April 2015).

20. Anh Nguyen, Jason Yosinski, and Jeff Clune. 2015. Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE.

21. Edmund T. Rolls. 1991. Neural organization of higher visual functions. *Current Opinion in Neurobiology* 1, 2 (1991), 274–278. DOI: http://dx.doi.org/10.1016/0959-4388(91)90090-T

22. Jimmy Secretan, Nicholas Beato, David B. D Ambrosio, Adelein Rodriguez, Adam Campbell, and Kenneth O. Stanley. 2008. Picbreeder: Evolving Pictures Collaboratively Online. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, Florence, Italy, 1759–1768. DOI: http://dx.doi.org/10.1145/1357054.1357328

23. Thomas Serre, Gabriel Kreiman, Minjoon Kouh, Charles Cadieu, Ulf Knoblich, and Tomaso Poggio. 2007. A quantitative theory of immediate visual recognition. In *Computational Neuroscience: Theoretical Insights into Brain Function*, Trevor Drew Paul Cisek and John F. Kalaska (Eds.). Progress in Brain Research, Vol. 165. Elsevier, 33–56.

24. Kenneth O. Stanley. 2006. Exploiting Regularity Without Development. In *Proceedings of the 2006 AAAI Fall Symposium on Developmental Systems*. AAAI Press, Menlo Park, CA.

25. Kenneth O. Stanley. 2007. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines* 8, 2 (2007), 131–162. DOI: http://dx.doi.org/10.1007/s10710-007-9028-8

26. Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *CoRR* abs/1312.6199 (2013).

27. Y. Taigman, Ming Yang, M. Ranzato, and L. Wolf. 2014. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. 1701–1708. DOI: http://dx.doi.org/10.1109/CVPR.2014.220

28. Xin-She Yang and Suash Deb. 2009. Cuckoo Search via Lévy flights. In *Nature Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. 210–214. DOI: http://dx.doi.org/10.1109/NABIC.2009.5393690

29. Xin-She Yang and Suash Deb. 2010. Engineering Optimisation by Cuckoo Search. *ArXiv e-prints* (May 2010).

30. Xin-She Yang and Suash Deb. 2014. Cuckoo search: recent advances and applications. *Neural Computing and Applications* 24, 1 (2014), 169–174. DOI: http://dx.doi.org/10.1007/s00521-013-1367-1