Genetic Deep Learning for Lung Cancer Screening

H. Michael Park and Connor Monahan

Innovation Dx Inc.

August 23, 2017

Abstract

Convolutional neural networks have shown great promise to improve the current state of medical imaging with computer vision. Using computer vision, it may be possible to make diagnoses on conventional chest X-rays (CXR) that are definitively apparent on subsequent CT and or biopsy. This may reduce the need for further evaluation with invasive testing or prevent errors of missed diagnoses as a second reader in today's fast paced and high volume clinical environment. Using over twelve thousand biopsy proven true positive cases of lung cancer from the Prostate, Lung, Colorectal and Ovarian (PLCO) dataset we trained a deep learning algorithm to predict the presence or absence of lung cancer within a CXR. Our classification algorithm achieved an accuracy of 96.00% with a PPV of 99.11% and a NPV of 93.25%

1 Introduction

The rate of lung cancers among the population has dramatically increased during the 20^{th} century, posing a large threat to the public especially those with higher risks to the disease, such as former or current smokers and those with exposure to radiation or chemicals in the workplace. It is estimated that over 1,465,000 people die every year from cancers, 18.2% of which are a variant of lung cancer [1]. The tumors resulting from this disease at a certain stage are visible to experienced radiologists on such mediums as chest x-rays, CT scans, and PET scans, imagery often taken during diagnosis of these diseases or as a preventative measure in several cases. Currently, the low survival rate for lung cancer of approximately 10% is attributed to its frequent late detection.

Over the past several years, image classification and object detection have seen tremendous improvements in performance as well as speed thanks to continued research into deep learning and convolutional neural networks (CNNs) [20]. The success and improvement found from deep learning is not from improved hardware and more encompassing datasets, but from innovations into model structure. From convolutions into fully connected layers [16], to the addition of dropout layers [11], to optimization [14, 2], the approach to deep learning is constantly changing and improving.

2 Background

2.1 Genetic Algorithms

Genetic algorithms (GA) were first introduced by John Holland in the early 1970s. Such algorithms are search heuristics that try to mimic the process of natural selection for the purpose of finding possible solutions to optimization as well as search problems. GAs have two main components. The first is a genetic representation of the solution space. In our case, that requires the ability to form every possible CNN architecture through our genetic encoding scheme. The second aspect of GAs is the ability to evaluate the fitness of solutions. We evaluate the fitness of a solution by first training it on the our training data, and then testing the trained solution on our test set, which then becomes the solution's fitness.

GAs begin with an initial population of genes and a population of solutions to the problem. Once every solution in the population has been evaluated, they are selected based on their fitness to be modified to create a new population (generation) of solutions, the more fit a solution, the more likely it is it have offspring. As generations are evaluated, the population becomes better at solving the task; the hope is that at least one individual across all generations was a good solution to the problem.

Evolutionary algorithms do not typically receive a lot of publicity, but in the past few years there have been isolated occurrences of people using genetic algorithms to solve nontrivial tasks such playing Mario by evolving basic neural networks [10] to build order optimization in StarCraft [5].

2.2 Advancements in Convolutional Neural Networks

Convolutional Neural Networks date back to 1989 with LeNet-5 [16]. LeNet-5 was designed to be able to handle a variance among identically labeled data by using "local receptive fields, shared weights, and spatial sub-sampling." Though the effectiveness was impressive, convolutional neural networks would not see the spot light again until AlexNet [15], which outlined how CNNs were used to win the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2012. After AlexNet, the trend of CNNs in literature was making CNNs linear, and deep. Two recent papers challenge this assumption and prove that CNNs are even more complex than had been thought.

The first is from ILSVRC 2014's winner, GoogLeNet [24]. GoogLeNet introduced the inception module. This inception module was designed with two ideas in mind: the approximation of a sparse structure with spatially repeated dense components and to keep the computational complexity in check. Unsurprisingly, the inception module does both of these exceptionally well; GoogLeNet used twelve times less parameters while still outperforming the winner of ILSVRC in 201, ALexNet. Inception architectures take advantage of the fact that convolutional filters of different sizes may extract features from separate clusters of information, and when one repeats them spatially they approximate said optimal sparse structure with dense components. The stacking of different sized convolutional filters concurrently allows for a significant increase in the number of units without an uncontrolled blow-up in computational complexity. This use of dimension reduction allows for shielding the large number of input filters of the last stage to the next layer. Inception architectures follows the principle that visual information should be processed at various scales and then aggregated so that the next stage can extract features from different scales simultaneously. The end result of the inception module is that they are up to three times faster than similarly performing networks with non-inception architecture [24].

The other major advancement in model design came from ILSVRC 2015's winner, Deep Residual Learning [9]. They asked themselves *"Is learning better networks as easy as stacking more layers?"* They discovered that when deeper networks begin converging, the network's accuracy becomes saturated, and then degrades rapidly. They wanted to ensure that a deeper model should produce no higher training error than its shallower counterpart, so to ensure this, they added identity mappings because solvers often struggle approximating identity mappings by multiple nonlinear layers. Residual learning ensures that solvers can easily approximate identity mappings by simply driving the weights of the multiple nonlinear layers toward zero to approach identity mappings.

Even with these breakthroughs in design techniques, it is still time consuming to hand design models. Recently, Google Brain released a paper outlining the usage of recurrent neural networks to search for optimal architectures using reinforcement learning [25]. The recurrent network acts as the controller, which generates a string encoding of a "child network". The child network is then trained on the real data, and the accuracy on the validation set is used as a reward signal for the controller. This reward signal is used to compute the policy gradient to update the controller, resulting in the next iteration's controller giving higher probabilities to architectures that perform better, thus improving the controller's search over time. This process achieved very strong empirical results on challenging benchmarks.

2.3 The Use of CNNs in Medical Applications

Using image analysis on medical images is not new to doctors or computer science. However, many applications of computer vision for medical applications struggle due to the medical data being noisy, inexact, sparse, or simply too big.

In a recent project performed at the University of Bern, a group of researchers created a deep convolutional neural network architecture for the classification of lung diseases based on lung slices from computed tomography images that performed with an accuracy of 85.5% on its data set [4]. They were able to successfully classify lung CT image patches between 6 different lung diseases. Their dataset consisted of 2,032 different diseases. To handle such a diverse dataset, they balanced their classes using a tree-taxonomy. Extremely small training classes is a major issue in applying deep learning to medical tasks, where rare diseases occur in less than one in ten thousand cases. To do this, they had a dynamic tree taxonomy and generated classes based on the number of examples rather than final diagnoses. This resulted in have 757 classes, instead of 2,032.

In another project performed at the Federal University of Parana, a convolutional neural network was used for classifying images of cell slides of breast cancer patients [22]. Using the BreaKHis dataset [21], they used AlexNet to classify microscopic biopsy images of benign and malignant breast tumors. Each slide of breast tissue contains 4 images, each with different levels of magnification. To handle the high resolution nature of their dataset, they invoked a few techniques. The first is the use of sliding windows with 50% overlap and the second is random crops of the raw image with no overlap. Interestingly enough, one cropping technique was not superior in the task of classification for all levels of magnification of the biopsy.

In [7], they achieve human specialist performance in several tasks. Their model was capable of identifying benign or malignant skin disease, and distinguishing high level taxonomic classes to identify lesions with similar treatment paths. In addition to this, they also achieve human level capability in specifically discriminating between treatment paths for lesions: biopsy/treat or not. For their training, they used transfer learning from imagenet, despite the image space of imagenet and their dataset being extremely different. This use of transfer learning may prove to be very

effective for other medical tasks.

3 Genetic Deep Learning

For this paper, we modified the NEAT algorithm to evolve the architecture of CNNs, dubbed "DeepNEAT", which we call DeepNEAT-Dx in [17]. More specifically, we injected convolution and pooling layers with pseudorandom hyperparameters into a minimal architecture and then optimized the weights through backpropagation on the training set. The fitness of a model is the final accuracy on the test set after 2 epochs of training.

There have been many proposed methods as to how Topology and Weight Evolving Artificial Neural Networks (TWEANNs) should be implemented and evolved [3, 6, 18]. However, in deep learning there are often times millions of parameters that require optimization, and doing so through any search algorithm is simply impractical, especially one based on chance.

A very critical topic when designing a genetic algorithm is what encoding scheme one should use. This has been an on going topic of discussion as encoding schemes directly contribute to the success of genetic algorithms. The requirements for our problem were as follows: the encoding had to be able to encode directed acyclic graphs (DAGs) of variable size, be able to track topological innovations over time, be able to easily create a new, coherent, individual from the genes of two parent individuals, and lastly the encoding scheme must inherently allow for efficient search for optimal network architectures.

With these requirements, there are only a few encoding schemes that are able to meet all these requirements. The performance difference between direct and indirect encoding depends largely on the problem [12]. Following in our predecessor's footsteps in [23], we chose direct encoding in the form of graph encoding. More specifically, we used Schiffman encoding [19]. Its basic structure is a list of neurons with their connectivity information. This allows us to easily program our own rules for mutations so mutations do not result in illegal phenotypes. Each vertex in the graph represented a layer in a CNN, and also stored hyper parameter information for the node it would construct.

We slightly modified the NEAT algorithm to be able to evolve CNNs. First, some basic mutations must be defined. *Inject Node* injects a random node (convolution, pool, or ReLU) with psuedo-random hyper parameters into the genome's network between a preexisting connection. Before injecting the new node, we checked to ensure that it would produce a valid network. If it did, the injection occurred, if it didn't, we changed the hyperparameters to values that would result in a valid network. This solves the problem of convolving the image to zero dimensions, as is guaranteed to occur as the number of injects increases. *Inject Segment* injects a pair of convolution with a ReLU, as well as a pool layer in a preexisting connection. *Point Mutate* changes the essential hyperparameters of a node. These basic mutation allows for most standard CNN architectures that one saw in literature before GoogleNet to be able to be constructed, such as AlexNet [15]. In addition to mutations, we incorporated speciation as outlined in [23], to allow for topological innovations to occur and prevent one topology framework from dominating the population.

3.1 Dataset

The dataset used for training and testing was compiled from the Prostate, Lung, Colorectal, and Ovarian Cancer Screening Trial (PLCO) dataset. This trial, organized by the National Cancer Institute, was a randomized, controlled trial to determine whether certain screening exams reduce the mortality of prostate, lung, colorectal, and ovarian cancer. Approximately 155,000 participants

were enrolled in the screening portion of the trial from 1993 to 2006. If a participant developed cancer at any point during the screening phase, all CXRs preceding the diagnosis were considered to be cancerous and therefore were marked as positives, which resulted in over 9,200 positives in our dataset.

The PLCO image dataset occupied 2.2TB in TIF format with individual CXRs having an approximate size of 2000x3000 pixels. In [8], their results were improved through down-scaling and then cropped. Our dataset was uniformly downscaled to 256 x 256 pixels and stored in PNG format. The dataset was randomly split 70% and 30% for training and testing.

3.2 Training

When training, we used a gene pool of 50 individuals, over the course of 10 generations. The genetic algorithm had mutation rate parameters that determined how often each mutation occurred. Each mutation received its own rate:

Mutation	Rate
Inject Convolution	50%
Inject Pooling	50%
Add ReLU	30%
Point Mutate	45%
Inject Segment	15%

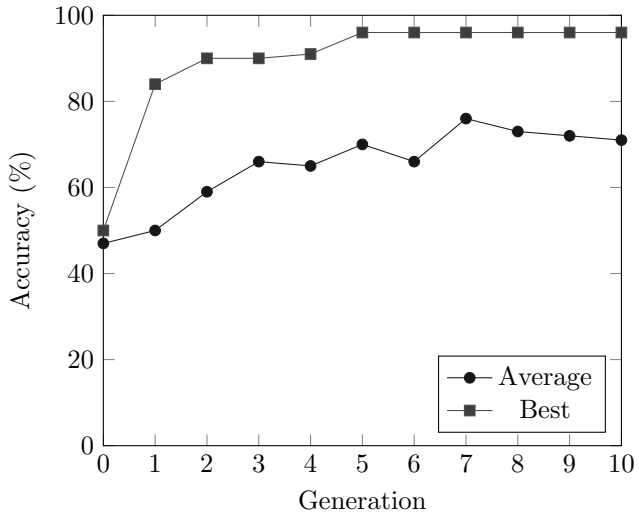
After the genetic algorithm produces a graph encoding of a network, we export the graph into a Caffe model [13]. Each model received identical training parameters, and they are as follows:

Hyper Parameter	Value
Solver	SGD
Epochs	5
LR Policy	INV
Base LR	0.01
Power	.75
Momentum	.90
Weight Decay	0.0005

3.3 Results

We ran several experiments with various population sizes and generations, and found that a evolving 50 individuals for 10 generations was the most efficient in producing a fit population quickly, as well as producing very fit top models. Unlike [25], which trained 12,800 models to optimize the RNN controller, we trained only 500 models and outperformed the state of the art classification models by 4%.

Figure 1: Improvement of Accuracy over Generations



To test the effectiveness of DeepNEAT-Dx, we compared it to the best models from the past 5 years. As shown in the table below, DeepNEAT-Dx surpasses every model in accuracy. Adding to this, DeepNEAT-Dx executed in its entirety in 4 hours, whereas ResNet-151 and GoogLeNet took over 40 and 35 hours to train, respectively.

Model	Accuracy
AlexNet	79.88%
GoogLeNet	89.34%
ResNet-50	90.22%
ResNet-100	90.68%
ResNet-151	92.03%
DeepNEAT-Dx	96.00%

For binary classifiers, such as this model, a matrix can be used to present several statistics about the performance of the algorithm. The core of this matrix is the contingency table, which is used to present the frequency of the real condition variable and the predicted condition variable. From this cross-tabulation many ratios can be derived, most notably the rates of false positives and negatives among the population. To generate the contingency table in Table 1, the network was computed against all 1,884 validation samples and the ground truth label was recorded with the predicted output. The statistical ratios were calculated from these inputs. The best model generated by our algorithm achieved an accuracy of 96.00% with a PPV of 99.11% and a NPV of 93.25%.

		Predicted			
	Population: 2627	Positive: 1235	Negative: 1392	Prevalence: 50.17%	
Confirmed	Positive: 1318	True Positive: 1224	False Negative: 94	True Positive Rate: 92.87%	False Negative Rate: 7.13%
	Negative: 1309	False Positive: 11	True Negative: 1298	False Positive Rate: 0.84%	True Negative Rate: 99.16%
	Accuracy: 96.00%	Positive Predictive Rate: 99.11%	False Omission Rate: 6.75%	Positive Likelihood Ratio: 11051.29%	Diagnostic Odds Ratio: 1536.51
		False Discovery Rate: 0.89%	Negative Predictive Value: 93.25%	Negative Likelihood Ratio: 7.19%	

Table 1: Binary classifier evaluation contingency table and confusion matrix

4 Discussion

The final accuracy of the network over the validation set of 96% shows that the model has succeeded at learning features related to the presence of various types of confirmed lung cancer in these images. Given that a human radiologist must spend a considerable amount of time on each image to make a correct prediction, screenings for many types of cancer do not often occur early, causing diagnoses to often arrive during the late stages of these diseases. This model can process images through at a speed of 3.41 milliseconds each, the potential to use such a model as a preliminary screening step may save many lives with early detection and from misdiagnoses. All cases of concern will necessarily have to be verified by an experienced radiologist, but the automation provided by this tool will decrease costs, increasing the accessibility of screenings, as well as increase the speed and accuracy of diagnoses.

Acknowledgments

This work would not have been possible without the support from the National Institute of Health and the data they provided us.

References

[1] Valentina Ambrosini et al. "PET/CT imaging in different types of lung cancer: An overview". In: *European Journal of Radiology* (2012), pp. 998–1001.

[2] Marcin Andrychowicz et al. "Learning to learn by gradient descent by gradient descent". In: *CoRR* abs/1606.04474 (2016). URL: `http://arxiv.org/abs/1606.04474`.

[3] Peter J. Angeline, Gregory M. Saunders, and Jordan B. Pollack. "An Evolutionary Algorithm that Constructs Recurrent Neural Networks". In: *IEEE TRANSACTIONS ON NEURAL NETWORKS* 5 (1994), pp. 54–65.

[4] M. Anthimopoulos et al. "Lung Pattern Classification for Interstitial Lung Diseases Using a Deep Convolutional Neural Network". In: *IEEE Transactions on Medical Imaging* 35.5 (May 2016), pp. 1207–1216. ISSN: 0278-0062.

[5] Jason M. Blackford and Gary Lamont. "The Real-Time Strategy Game Multi-Objective Build Order Problem". In: AIIDE, 2014.

[6] D. Dasgupta and D.R. McGregor. "Designing application-specific neural networks using the structured genetic algorithm". In: IEEE, 1992.

[7] Andre Esteva et al. "Dermatologist-Level Classification of Skin Cancer With Deep Neural Networks". In: *Nature* (2017), pp. 115–118.

[8] L. G. Hafemann, L. S. Oliveira, and P. Cavalin. "Forest Species Recognition Using Deep Convolutional Neural Networks". In: *2014 22nd International Conference on Pattern Recognition*. Sept. 2014, pp. 1103–1107. DOI: 10.1109/ICPR.2014.199.

[9] Kaiming He et al. "Deep residual learning for image recognition". In: Computer Vision and Pattern Recognition, 2016.

[10] Seth Hendrickson. *MarI-O*. https://github.com/pakoito/MarI-O. 2015.

[11] G. E. Hinton et al. "Improving neural networks by preventing co-adaptation of feature detectors". In: *arXiv preprint arXiv:1207.0580* (2012).

[12] Sumati jaggi, Ravi Bhushan, and Davender Malhotra. "Guidelines to Decide the Encoding Scheme Used For G.A." In: *International Journal of Advanced Research in Computer Science and Software Engineering* Vol. 3, Issue 8 (2013), pp. 1436–1440.

[13] Yangqing Jia et al. "Caffe: Convolutional Architecture for Fast Feature Embedding". In: *arXiv preprint arXiv:1408.5093* (2014).

[14] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980 (2014). URL: http://arxiv.org/abs/1412.6980.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks". In: *Neural Information Processing Systems* (2012).

[16] Yann LeCun et al. "Gradient-Based Learning Applied to Document Recognition". In: *Proceedings of the IEEE* Vol. 86, Issue 11 (1998), pp. 2278–2324.

[17] Risto Miikkulainen et al. "Evolving Deep Neural Networks". In: *CoRR* abs/1703.00548 (2017). URL: http://arxiv.org/abs/1703.00548.

[18] Joao Carlos Figueira Pujol and Riccardo Poli. *Evolving the Topology and the Weights of Neural Networks Using a Dual Representation*. 1998.

[19] Wolfram Schiffmann, Merten Joost, and Randolf Werner. "Performance Evaluation of Evolutionary Created Neural Network Topologies". In: *Springer-Verlag London* Vol. 2 (1990), pp. 274–283.

[20] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556 (2014). URL: http://arxiv.org/abs/1409.1556.

[21] F. A. Spanhol et al. “A Dataset for Breast Cancer Histopathological Image Classification”. In: *IEEE Transactions on Biomedical Engineering* 63.7 (Aug. 2016), pp. 1455–1462. ISSN: 0018-9294. DOI: 10.1109/TBME.2015.2496264.

[22] F. A. Spanhol et al. “Breast cancer histopathological image classification using Convolutional Neural Networks”. In: *2016 International Joint Conference on Neural Networks (IJCNN)*. Aug. 2016, pp. 2560–2567. DOI: 10.1109/IJCNN.2016.7727519.

[23] Kenneth O. Stanley and Risto Miikkulainen. “Evolving neural networks through augmenting topologies”. In: *Journal of Evolutionary Computation* Vol. 10 Issue 2 (2002), pp. 99–127.

[24] Christian Szegedy et al. “Going deeper with convolutions”. In: *Computer Vision and Pattern Recognition* (2015).

[25] Barret Zoph and Quoc V. Le. “Neural Architecture Search with Reinforcement Learning”. In: *CoRR* abs/1611.01578 (2016).