AUTOMATED DESIGN USING NEURAL NETWORKS AND GRADIENT DECENT

Oliver Hennigh
Mexico
Guanajuato, Gto, Mexico
loliverhennigh101@gmail.com

ABSTRACT

Automated Design is the process by which a object is designed by a computer to meet or maximize some measurable objective. In this paper, we propose a novel method that makes use of deep neural network and gradient decent to perform automated design on complex real world task. Our approach works by training a neural network to mimic the fittness function of the optimization task and then, using the differential nature of the neural network, we perform gradient decent to maximize the fittness. We demonstrate this methods effectiveness by designing an optimized heat sink and both 2D and 3D wingfoils that maximize the lift drag ratio under steady state flow conditions. We highlight that our method has two disticnt benifits. First, evaluating the neural networks prediction of fittness can be orders of magnatude faster as is the case with fluid flow problems. Second, using gradient decent allows the design space to be search much more effiently.

1 INTRODUCTION

Automated Design is the process by which a object is designed by a computer to meet or maximize some measurable objective. This is typicaly performed by modeling the system and then exploring the space of designs to maximize some desired fittness function whether that be automotice car styling with low drag or an integrated circuit with low profile. The most notable historic example of this is the 2006 NASA ST5 spacecraft antenna designed by a evolutionary algorithm to create the best radiation pattern. More recently there has been (Flow Sculpter) and (optical computer). While there have been some significant sucesses in this feild the dream of true automated is still far from realized. The main challanges faced are heavy computational requirements for accuratly modeling the physical system under investigation and often exponentialy large search space. These two problems negatively complement eachother making the computation requirements intractable for even simple problems. For example, in the realively simple flow problems explored in this work, a heavily optimized flow solver running on modern gpus requires around 5 minutes to perform each simulation. Given that as many as 3,000 designs need to be tested to acheive reasonable performance, this results in a total computation time of 9 days on a single GPU. Increassing the resolution of the simulation or expanding the parameter space quickly make this an unrealizable problem without considerable resources.

Our approach works to solve the current problems of automated design in two ways. First, we learn a computationaly effeicent representation of the physical system on a neural network. This trained network can be used evaluate the quality or fittness of the design several orders of magnatude faster. Second, we use the differentiable nature of the trained network to get a gradient on the parameter space when performing optimization. This allows significanly more efficient optimization requiring far fewer iterations then other methods. These two abilitys of our method overcome the present difficulties with automated design and allow the previous mention 9 day optimization to be run in only 10 mins. While we only look at two relavively simple problems in this work, we enphasize that the ideas behind our method are applicable to a wide variety of automated design problems.

The first problem tackled in this work is designing a simple heat sink to maximize the cooling of a heat source. The setup of our simulation is ment to somewhat minimc the conditions seen in a heat sink on a computer processor. We keep this optimization problem realatively simple though and use this as a first test and introduction to the method.

We also test our method on the significantly more difficult task of designing both 2D and 3D wing-foil with high lift drag ratios under steady state flow conditions. This problem is of tremendous inportance in many engineering areas such as aeornotical, aerospace and automotive engineering. Because this is a particularly challanging problem and often times unintuative for designers, there has been considerable work using automated design to produce optimized designs. We view this as true test our our method and both its advantages and disadvantages.

As we will go into detail in later section, in order to perform our flow optimization tests we need a network that predicts the steady state flow from an objects geometry. This problem has previously been tackled here where they use a relatively simple network architeture. We found that better perform could be obtained using some of the modern network architecture developments. For this reason, in addition to presenting our novel method of design optimization, we also present this superiour network for predicting steady state fluid flow with nerual networks.

This work has the following contributions.

- We demonstrate a novel way to use neural networks to greatly accelerate automated design problems.
- We present this method in such a way that it can ready applied to many other automated design problems.
- We new network architeture for predicting steady state fluid flow that vastly out performs previous models.

2 Related Work

This is touches on several vastly diffrent subjects. Because of this we have provided a breif review of the related areas.

2.1 Neural Network based Surragate models

Our method uses a network to predict steady state fluid flow from boundary conditions. This idea was first presented in (Guo et al., 2016) where they showed. Our flow prediction network has several key difference to this original work. First, we heavily improve the network architecture by keeping the network all convolutional and taking advantage of both residual connections and a U-Network architeture. This proved to drasticaly improve accuracy while maintain fast computation. Second, it takes in the binary representation of the boundary conditions instead of the Signed distance map. We found that with our improved network architecture, we over came the issure in their work using such a representation of the boundary.

2.2 Automated Design Optimization of Airfoils

To data, there has been substantial work in automated aerodynamical design for use in aeronotical and automotice applications. The standard work flow is to first parameterize the search space of designs then iteratively simulate and evaluate the. We use the A veriety of search methods have been used with some sucess in the optimization process. Simulated Annealling, Genetic Algorithms and Particle swarm have all been used with vering degrees of sucess.

Automotive Aerodynamic Design Exploration Employing New Optimization Methodology Based on CFD

Adaptive Shape Parameterization for Aerodynamic Design

Pros and Cons of Airfoil Optimization 1

Multi-level CFD-based Airfoil Shape Optimization With Automated Low-fidelity Model Selection

A Surface Parameterization Method for Airfoil Optimization and High Lift 2D Geometries Utilizing the CST Methodology

A Universal Parametric Geometry Representation Method CST

3 Gradient Decent on Parameter Space

Our optimization problem can be viewed in concrete terms as

Our method We look at the two network architetures of inter and explain how they can be used in tandum perform boundary optimization.

3.1 Flow Prediction Network

As mentioned above, we have made significant changes to prior network architecture design . A figure illustrating our networn can be found here. This model resembols the u-network architeture seen here with a series skip after each down sample. The advantages of this style of network are its high performance on image to image type tasks, trainablility on small datasets, and fast evaluation time in comparison to networks with large fully connected layers. The trainability on small datasets make this particulary effective on predicting steady state flow. Creating datasets of flow simulation to train on can be one of the most difficult aspects of this work and requires considerable computation. Our network is able to train on realively small datasets of only 4,000 flow simulation in comparision to the 100,000 required in previous work predicting steady state flow.

In this work we reley heavely on the previous work paramertizing wingfoils. We use the

4 Experiments

In the following sections we subject our method and model to a variety of test. The goal the goal being first to determine how accuraty and fast our model can predict steady state flow. In particular, what its accuracy in predicting values such a drag and lift as these are important quantitize in our optimization. Second, we investigate how effective our gradient decent optimization is. This line of tests compares our method to other none gradient based search techniques and illustrates what is happening in the optimization processes looks like. For example, what does the gradient surface look like.

4.1 Flow Prediction Accuracy

As mentioned before, our model significantly out performs priour work predicting steady state fluid flow.

`../test/figs/generated_flow_difference.jpeg`

Figure 1: Flow

`../test/figs/flow_accuracy_2d.jpeg`

Figure 2: Flow

`../test/figs/boundary_space_explort.jpeg`

Figure 3: Flow

4.2 Automated Design of 2D and 3D Airfoils

5 Conclusion

In this work we hav

Table 1: Sample table title

Batch Size	1	2	4	8	16
Flow Net 512^2	0.150 sec	0.101 sec	0.077 sec	0.065 sec	0.058 sec
Param Net 512^2	0.083 sec	0.045 sec	0.026 sec	0.015 sec	0.011 sec
Learn Step 512^2	0.494 sec	0.345 sec	0.270 sec	0.231 sec	Nan
Flow Net 144^3	0.826 sec	0.686 sec	0.627 sec	0.623 sec	Nan
Param Net 144^3	0.195 sec	0.144 sec	0.119 sec	0.106 sec	0.093 sec
Learn Step 144^3	3.781 sec	Nan	Nan	Nan	Nan

6 Sample Tables

Acknowledgments

This work was made possible through the `http://aigrant.org` run by Nat Friedman and Daniel Gross. This work would not have be possible without this very genourous support.

References

Xiaoxiao Guo, Wei Li, and Francesco Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 481–490. ACM, 2016.