THE OHIO STATE UNIVERSITY
Department of Physics

Big Data Final Project

FALL, 2019

R. HUGHES

Course Final Project

My hope for this course is that it will give you practical data analytics tools to use, if and when you need them, whether in a future research project or as a data scientist in industry. The Final Project is a chance for you to demonstrate that you have learned enough to handle a machine learning task - on your own!

Goals:

1. The project should be one in which machine learning/advanced algorithms can be brought to bear on an interesting problem, either to "solve" that problem, or at least deliver meaningful insights regarding the problem.
2. The projects will all be single person projects.
3. The projects should be science oriented.
4. The deliverables do not include a final report but instead a working jupyter notebook as well as a single page “Poster” (in ppt/pdf form only).

Project Choices:

You have the choice of attempting a predefined project or coming up with your own. In either case, **it will be necessary that the project has some reasonable connection to science.** I am comfortable with the possibility that this connection may not be obvious, but if so you will need to make the case in your project proposal that such a connection with science exists.

1. **Already existing project:** For these you can use the resources of the Kaggle website (or other similar online resources). Kaggle hosts open machine learning competitions for both commercial and non-profit organizations. They provide data, background information, an evaluation metric, and a forum for each competition. There are (as of Oct 8, 2019) 14 open competitions and 341 completed competitions. You can choose an open OR an already completed competition. I anticipate that you may need to substantially modify the goals as stated on the Kaggle website in order to be able to complete your project in the allotted time.

2. **You choose the project:** Here you are free to choose something from current research that you are doing, or perhaps something you have been interested in learning about. You are also allowed to combine this project with a project from another class, though you must fulfill all requirements of this project. More on this below in the FAQ.

Machine Learning Project Checklist

1. Define the problem: what exactly are you trying to do?
2. Get the data. (Sometimes this means creating your data!)
3. Explore the data with an eye to learning how you might use the data to solve your problem.
4. Prepare the data: this may include normalizing, augmenting, modifying (cropping, centering, etc) the data.
5. Investigate a number of different models and choose 2 or 3 that seem particularly good.
6. Fine tune the models and either choose the best or combine them.
7. Present your solution to the problem using your final model.
8. Launch your system with your final model and monitor results.

Project Timeline

1. **Proposal:** This describes the project that you will do, as well as outlining your proposed approach. This approach may change as you learn more!
   - **Due date: 5pm Thursday October 31, 2019**
2. **Mid-term "Report":** This is a checklist of things you have managed to complete, as well as a preliminary version of your poster (see below). The primary purpose of this is to make sure you are on track to complete the project.
   - **Due Date: 5pm Thursday November 14, 2019**
3. **Completed project:** A completed project will be comprised of a jupyter notebook(s) as well as a final poster in pdf/ppt form, all residing in your github folder:
   - **Due Date: 5pm Monday December 9, 2019**

Project Proposal

This describes the project you will do, as well as outlining your proposed approach. This approach may change as you learn more! Sources for possible projects are listed.

* Due date: **5pm Thursday October 31, 2019**

* Deliverables: Completed proposal form; see the file "Project Proposal Template.doc" in the finalProject/docs folder in your github repo.

* Grading: This is a completion based grade, worth **10% of the project points**, if all parts are done and it is turned in on time. You will lose 25% (of the 10%) for each day it is turned in late. I encourage you to share with me your proposed project ahead of the due date if you have any concerns that it is not appropriate.

Project Proposal

[Name]
[Date]

Project Proposal: [Name of Project]

Project Source
[Kaggle/UCI/Other Online/Research]

Background
[Why is the project being undertaken? Describe an opportunity or problem that the project is to address.]

Objectives/Goals

- [specific & measurable objective 1]
- [specific & measurable objective 2]
- [specific & measurable objective 3]

Proposed Timeframe

Description of Work	Start Date	End Date
Research on problem/previous approaches	October 31, 2019	
Obtain Data	…	
Visualize/Explore data		
Mid-project Checklist Due	November 14, 2019	
Begin Model Testing		
Fine-tune Model		
Prepare Poster		
Project Due	December 9, 2019	

Signatures

[Student Name] [Signature]

[Instructor Name] [Signature]

Mid-term "Report"

This is checklist of things you have managed to complete, as well as a preliminary version of your poster (see below). The primary purpose of this is to make sure you are on track to complete the project.

Due Date: **5pm Thursday November 14, 2019**

Deliverables: Completed checklist (see finalProject/docs folder in your github repo); associated items highlighted in checklist; current version of your poster (see finalProject/poster folder in your github repo). It is ok if most of your poster is placeholders at this point.

Grading: This is a completion based grade, worth **10% of the project points**. Completion means:

- the entire checklist is filled out
- At least 3 of the item in the checklist are marked as completed AND the associated jupyter notebooks are in the github folder

Mid-term "Report"

Mid-Project Report and Checklist

Name:	
Date:	
Project Title:	

Yes	No	Date Complete	Task
			Data has been transferred to the appropriate computing platform (OSC/research group/personal laptop)
			If not complete explain:
			Background research has been completed (and section for poster summerizing this is done).
			If not complete explain:
			Preliminary visualization of data have been performed. Jupyter notebook available in project folder
			If not complete explain:
			Timeline for attaining remaining project goals has been sketched out. Text doc available in project folder
			If not complete explain:
			Project poster has been started. Version available in project folder.
			If not complete explain:
			Initial/test model has been chosen and initial pass with this model has been attempted.
			If not complete explain:

Completed project: Due Date: 5pm Monday December 9, 2019

A completed project will be comprised of the following, all residing in your github folder:

1. An annotated jupyter notebook (one or more) which can be used to recreate your project. Some parts of the project may end up using bash scripts and full python files in addition to or instead of jupyter notebooks.
2. A final poster (in ppt *and* pdf form only - there is no formal poster session). The final poster should be in a form such that you could give a copy to an interested party (future employer or research advisor) to give them insight into the kind of work you are capable of.
3. Grading: A rubric will be available soon.

Resources

Here are some online sources:

1. The Kaggle competition [website](https://www.kaggle.com/competitions).
2. The UCI machine learning repository [website](http://archive.ics.uci.edu/ml/index.php)
3. The Stanford CS229 project [website](http://cs229.stanford.edu/projects.html). The section titled "Previous projects" has hundreds of example projects from past years. Note that posters only appear to be available for years 2015-present, and reports (something I am not asking you to do) are available for all years. Many (though not all) of these projects are also based on Kaggle competitions. Some particularly good examples of posters are:
   - Large-scale Protein Atlas Compartmentalization Analysis]
   - A Data-Driven Approach for Predicting Elastic Properties of Inorganic Materials]
   - Galaxy Morphology
   - Modeling and Optimization of Optical Devices using a Variational Autoencoder

FAQ(1)

1. I found on the web a complete working solution to my project problem. Can I use it?

No. As mentioned above, each Kaggle competition has a forum of questions and answers devoted to that particular competition. We encourage you to consult that forum! However, as much as is practical, we strongly urge you to **not** consult any posted solutions. This will defeat the whole purpose of the project: The value the project has as a learning experience is **much** more important than its contribution to your final grade in this course!

2. Can I do this project as part of a team with another student?

No. However, we do anticipate that many students in the class may end up working on the same subset of projects. Consulting with these students about problems you run into and methods you are trying is fine. But the work you submit as your project must be **overwhelmingly** your own. You should be able to explain and justify every choice that is made in the course of completing your project.

3. Is it okay to use a dataset that is not public ?

Yes. We don't mind you using a dataset that is not public, as long as you have the required permissions to use it. We don't require you to share the dataset either as long as you can accurately describe it in the Final Report.

FAQ(2)

4. Are the final presentations public?

Yes. All projects will be viewable by fellow students via our classroom github repo. We will also host final posters (just like the CS229 website above). Let us know if this is not possible (because the data is private) or desired for your project. We want to advertise this course and its potential to future students!

5. Can I combine this project with another class project?

Yes. In general it is possible to combine your project for this course and another class, but with the following caveats:

* You should make sure that you follow all the guidelines and requirements for this project (in addition to the requirements of the other class). So, if you'd like to combine this project with a class X but class X's policies don't allow for it, you cannot do it.

* You cannot turn in an identical project for both classes, but you can share common infrastructure/code base/datasets across the two classes.

* Clearly indicate in your mid-project report as well as an addendum (text document) to your final poster, which part of the project is done for this class and which part is done for a class other than this one. For shared projects, we also require that you submit the final report from the class you're sharing the project with.

Example project from 2018

PHY 6820/ Big Data Analytics/ OSU

Galaxy Morphology Classification

INTRODUCTION

Large scale sky surveys, such as The Sloan Digital Sky Survey (SDSS) (SDSS: York et al. 2000), have been able to provide astronomers with immense amount of astronomical and galactic data. The size of these data sets has increased the need to automate analysis, so astronomers have turned to Deep Learning to process and analyze these large data set.

One of the ways deep learning is being used is in galaxy morphology classification. This is because morphological features provide insights into the physical processes that shape the evolution of galaxies.

AIM

- Create a Convolutional Neural Network using *Keras* (deep-learning framework) to classify galaxies into three different categories
  1. **Class 1: Elliptical**
  2. **Class 2: Spiral**
  3. **Class 3: Other**

DATA SET

Galaxy Zoo images were obtained from the *kaggle galaxy challenge* dataset. A total of 61,578 pre-classified training images were used, each of size 424 x 424 in RGB

IMAGE PRE-PROCESSING

Image size was reduced to 212 x 212 in RGB, based on the fidings of Kabani et al. 2016.

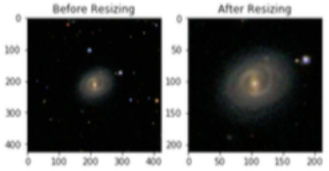


Figure 1. Galaxy image resized and centered

DATA AUGMENTATION

Out of the 61,578 images, there were surprisingly only 59 images classified as **other** which meant there was a lack of class 3 training/test data. To increase the data set, the *Keras data augmentation transformation* was used to create "new" images.

The following was done to each image:

- Rotated the image:
  - Range of 40 degrees for training set
  - Range of 60 degrees for test set
- Applied a horizontal and/or vertical flip

The **Train images** were augmented by a factor of 500, and the **Test images** by a factor of 118.

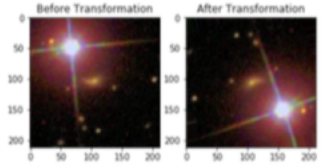


Figure 2. Transformed image from class 3

Convolutional Neural Network (CNN)

The CNN developed is shown in Figure 3.

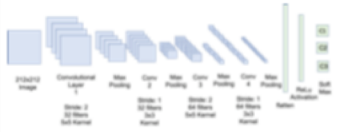
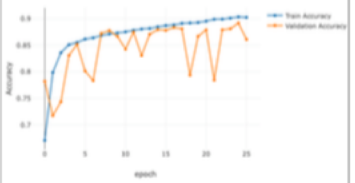


Figure 3. Convolutional Neural Network

The CNN developed consisted of 4 Conv2 layers of different strides and kernel sizes, all leading up to fully connected layers that would then output the three different classes of galaxies. The choice of 4 layers was picked after many trials along with the number of strides and kernel sizes.

Figure 4. CNN Accuracy with Train and Validation Images



TRAINING AND TEST IMAGES:

A split of 80% testing – 20% test was applied to each class before combining and shuffling the images. Recall that the class 3 images were augmented, therefore the total number of images used is larger than the original data set.

RESULTS:

Training and Validation Images: Accuracy and Loss

A total of 62,971 images were used for training and 15,743 images used as validation. The performance of the CNN resulted in a validation accuracy of 89%, which can be seen in Figure 4, and a loss of 0.27. This was calculated with a patience of 20 at epoch 24.

Test Images:

This "best CNN model" was then applied to 19,267 test images. To see how well our trained CNN classifies galaxies, the confusion matrix in Figure 5. was created.

		Predicted Class 1: Elliptical	Predicted Class 2: Spiral	Predicted Class 3: Other
True	Class 1: Elliptical	4515	818	6
True	Class 2: Spiral	1379	5569	18
True	Class 3: Other	371	96	6495

Figure 5. "Best" CNN model confusion matrix – Test Images

RESULTS - CONTINUED

The confusion matrix in Figure 5. shows that our mode has difficulty distinguishing between elliptical and spiral galaxies. To see by how much, precision and recall was calculated for each class.

	Class 1: Elliptical	Class2: Spiral	Class 3: Other	Averaged
Precision	0.721	0.859	0.996	0.858
Recall	0.846	0.799	0.933	0.859

Figure 5. Precision and Recall for test images

CONCLUSIONS

The deep plunges in the validation accuracy can be attributed to the lack of class 3 images. Each of these plunges can be thought of the network "memorizing" the data, suggesting that data augmentation may not be the best approach to handle data sets with very few samples.

The validation accuracy increased to 89% by adding only a few layers, and increasing the filter size, from 32 to 64, only once. Reducing the size of the image after the first convolution layer also helped, this was done by increasing the stride and size of the kernel.

Precision was lowest for elliptical and spiral galaxies, whereas it was the highest for other. This could just mean that even though the class 3 images were transformed and made to look like "new" images, some "memorization" was inevitable, and maybe image transformation does not change the image as much as we would like.

REFERENCES

1. York, Donald G., et al. "The sloan digital sky survey: Technical summary." The Astronomical Journal 120.3 (2000): 1579.
2. Willett, Kyle W., et al. "Galaxy Zoo 2: detailed morphological classifications for 304 122 galaxies from the Sloan Digital Sky Survey." *Monthly Notices of the Royal Astronomical Society* 435.4 (2013): 2835-2860.
3. Kabani, Abdul Wahab, and Mahmoud R. El-Sakka. "How Important is Scale in Galaxy Image Classification ?." *VISIGRAPP (4: VISAPP)*. 2016.

ACKNOWLEDGEMENTS

I'd like to thank Professor Hughes for his help in trouble shooting and brain storming with me when nothing seemed to work correctly.

THE OHIO STATE UNIVERSITY

Example project from 2018

College of Arts & Sciences
Department of Physics

Higgs Search Using Machine Learning

Overview

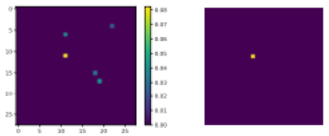
The application of machine learning techniques to particle physics analysis is an active and interesting area of research. Such techniques could obviate the need for lengthy data preparation while improving results.

Goals:

1. Train a five-layer Fully Connected Network (FCN) on all parameters and compare to results in [1]
2. Improve on previous results by means of a more complex network, acting on both low- and high-level parameters.
3. Examine the filters of the trained CNN to determine what features in the data for which its filters are searching.

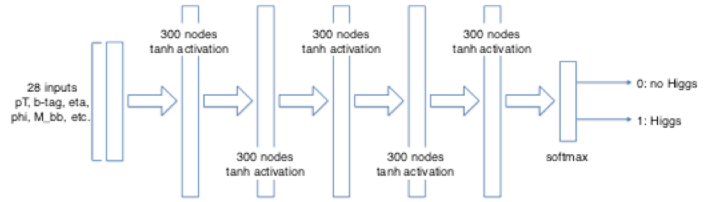
Data

- 1,100,000 Monte Carlo-simulated collider events
  - 500,000 used as test set
- 28 features
  - 21 are kinematic properties (detector outputs)
    - low-level parameters
    - E.g. pT, b-tag, eta
  - 7 are functions of low-level parameters
    - high-level parameters
    - invariant masses (M_bb, M_jj, etc.)
- Each event tagged as a "Higgs" or "no-Higgs" event
- Data organized into eta-phi map, with pT, b-tag and lepton data as layers (example below)

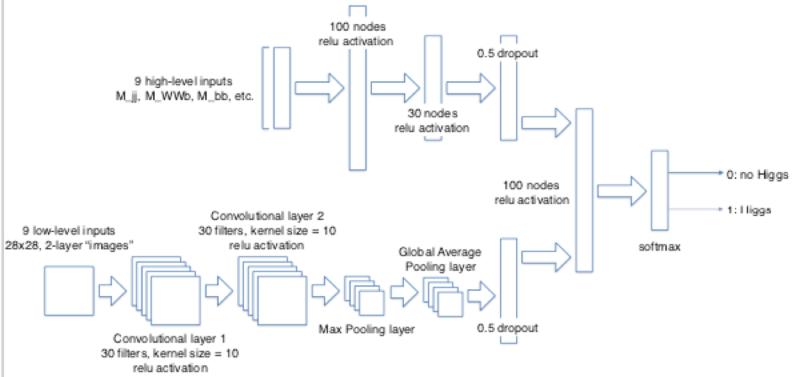


Models used

Two neural network architectures were explored:



- A fully connected 5-layer NN; all 28 parameters were used as inputs (460,802 trainable parameters)
- Matched architecture in [1]



- A two-headed network, consisting of a CNN for the low-level parameters and an FCN for the high-level parameters (106,392 trainable parameters)
- Fewer parameters = lower odds of overfitting

Model Results

Five-layer FCN (1,050,000 training events, 50,000 testing events

	Predicted Background	Predicted Signal
Background	167,780	60,405
Signal	67,695	204,102

- Precision = TP/(TP + FP) = 0.7716
- Recall = TP/(TP+FN) = 0.7509
- AUC = 0.8251

CNN (525,000 training events, 25,000 test events

	Predicted Background	Predicted Signal
Background	7508	2744
Signal	4202	10,546

- Precision = TP/(TP + FP) = 0.7935
- Recall = TP/(TP+FN) = 0.7151
- AUC = 0.7999

Conclusions

- Performance of the FCN was similar to that found in [1]
- The CNN produced similar, but still somewhat weaker results
  - Possible tweaks to structure could result in improved or even superior performance
- The CNN was trained on half of the data and a quarter of the trainable parameters to achieve the present results
  - Possible spatial dependence of the input parameters taken into account by the convolutional structure
  - Could point to underlying structure exploitable in future analysis

Citations

(1) Baldi, P. et al. Searching for exotic particles in high-energy physics with deep learning. Nat. Commun. 5:4308 doi: 10.1038/ncomms5308 (2014)

(2) Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

THE OHIO STATE UNIVERSITY

Example project from 2018

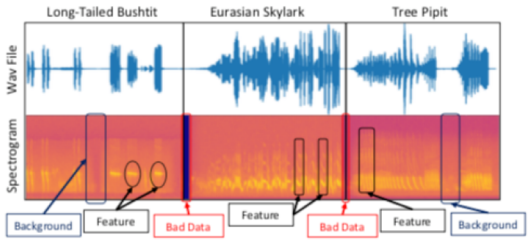
The Ohio State University Department of Physics Big Data Analysis

The ICML Bird 2013 Bird Challenge

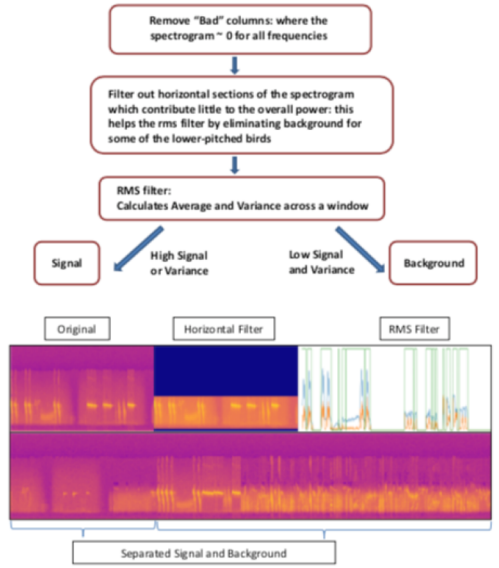
https://www.kaggle.com/c/the-icml-2013-bird-challenge

Identify Bird Species From Continuous Audio Recordings

Between the stages of condensed matter and plasma lies the poorly understood transition stage called warm dense matter (WMD). The short life span of WDM makes measurement-taking difficult. As well, the combination of high material density with ionizing temperatures makes traditional condensed mater and plasma physics behavioral prediction models hard to apply. New models and experimental validation of those models are necessary for the improvement of temperature-sensitive experimental physics such as ICF (inertial confinement fusion).

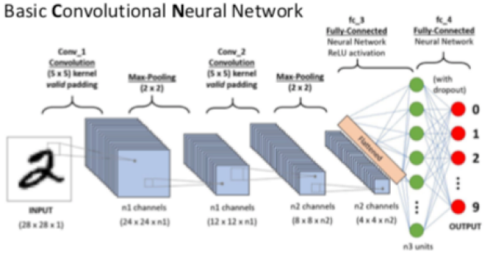


Splitting up Data Into Signal and Background:



Using a CNN as a Simple Binary Classifier on the Separated Signal and Background:

Basic Convolutional Neural Network



https://mc.ai/a-comprehensive-guide-to-convolutional-neural-networks

Network Used:
Input Images: Down-sampled to 128 x 64 pixels
CNN
- Convolutional 32, (5 x 5) filters, activation = relu
- Max-Pooling (2 x 2)
- Convolutional 32, (5 x 5) filters, activation = relu
- Max-Pooling (2 x 2)

Dense 100, activation = relu
Dense 2, activation = softmax

Total Training Samples: 40
Total Test Samples: 18

Results:
Validation Accuracy: 94%
Validation Loss: 0.25

Classifying All 35 Birds also using a simple CNN:

For this classification, it is notable that, after separating out the background and splitting the signal up into usable images:

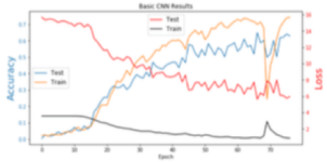
- The maximum number of labeled signal images from a bird was only 36, and the minimum was 12!
- These were further split into training and testing samples.
- If there were less than 36 samples (most birds), oversampling was used.
- Samples used for testing were completely different from samples used for training
- A background category was added by randomly sampling from all removed backgrounds

Network Used:
Input Images: Down-sampled to 128 x 64 pixels
CNN
- 3 X [Convolutional 32, (3 x 3), relu + Max-Pool (2x2)]
- Convolutional 64, (5 x 5), relu + Max-Pool (2x2)

Dense 30, activation = relu
Dense 36, activation = softmax

The number of Features in the 1st Dense Layer was chosen by making it a Tunable, Hyper-Parameter and testing in a range of 10 to 100

Accuracy and Loss Results (Testing/Validation)



Over %60 Validation Accuracy: This is much better than random picking, which would only be correct 1 in 36 or ~3% of the time.

Confusion Matrix:

For a multiclass-classifier such as this, it is more informative to look at the "confusion matrix" which specifies exactly how many (or what percent) of a given class were labeled correctly, or as some other class.
The following table gives a summary of the confusion matrix.

Correct Classifications: out of 10	Misclassified									
Bird 1	2	2	3	6	8	9	13	15	16	26
Bird 2	1	12	27	10	28	16	30	33	4	14
Wrong	4	4	9	4	5	4	7	6	4	5
Bird	0	1	2	3	4	5	6	7	8	9
Right	8	9	1	0	10	7	6	10	0	0
Bird	10	11	12	13	14	15	16	17	18	19
Right	5	8	8	0	10	0	0	7	9	10
Bird	20	21	22	23	24	25	26	27	28	29
Right	4	5	3	8	10	8	3	8	10	9
Bird	30	31	32	33	34					
Right	9	10	9	8	10					

Confused!

Possibilities for Improvement

The small amount of labeled data makes this project quite difficult overall, but there are quite a few tricks that could be tried in order to improve the classifier:

- Train and Auto-Encoder on *all* the data and use its "encoder" segment as the first layer of the classifier
  - Auto-Encoders learn the best ways to represent the data they are trained on rather than learn to classify the data. The AE can be trained on the overwhelming amount of unlabeled data, which would likely make it really good at creating good representations of the data. Then the encoder part could replace the CNN section of the classifier
- Augment the Labeled Data
  - Keras provides nice built-in methods for transforming the labeled data a bit in order to create "new" labeled data.
  - Many other interesting filters can be applied to the data in order to create new data, here is an example of transforming the spectrograms using a Librosa library "vocal filter"

- Make the parameters of the Signal/Background algorithm Tunable Hyper-Parameters and use the Binary Classifier to set them on a per-bird basis
  - The idea is that, if the classifier can achieve better accuracy at differentiating between the background and signal, the parameters used to decided which was which were better
- Many more possibilities exist than these!

Example project from 2018

PHYSICS DEPARTMENT

Denoising Short Time Exposure Images in Florescence Microscopy

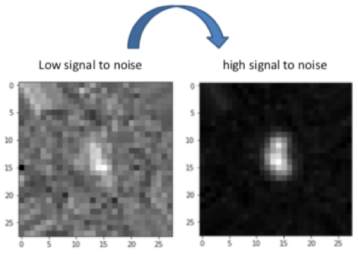
[redacted]
akl2@osu.edu

INTRODUCTION

Florescence microscopy is the most widely known technique for probing live cell behavior. Experimental discovery of new biology is limited by the technique used. Minimizing photo-bleaching and photo-toxicity, through, for example, short time exposure of light comes at the expense of low signal to noise ratio.

AIM

Through this project, I aim to correct that limitation computationally through a network that will be able to predict what the high signal to noise version of the image looks like. We do that using convolutional neural network autoencoder.



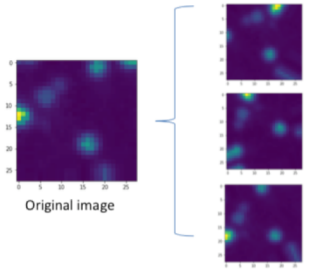
METHODS

Obtaining data

Through experiment, imaging the same field of view with 500 ms light exposure and with 50 ms light exposure for almost 300 times as to get a range of intensities from the cell. Obtained images are 1200 X 1200 pixels but later cropped to be 28 X 28.

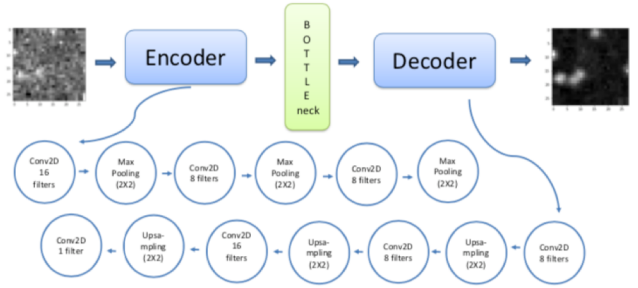
Data Augmentation:

Working with data sample size of 2300 is not enough to train the autoencoder so using Keras image preprocessing function we can generate an infinite amount of data through rotation and flipping.

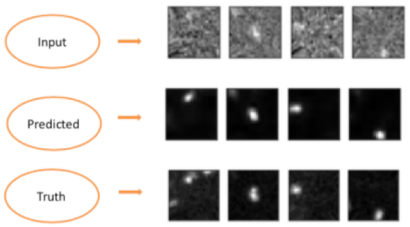


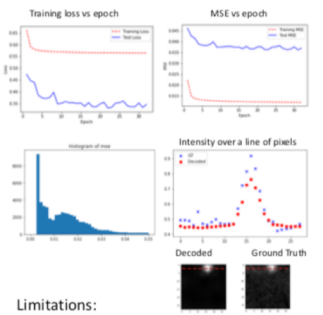
Network: Convolutional Neural Network Autoencoder

The network is trained on low SNR images and given the high ones as the target.



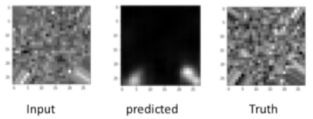
RESULTS





Limitations:

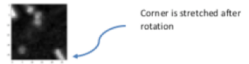
The model tries to create objects even when the input is in fact pure noise with no signal. This, I think, is due to "fake" structures caused by the image rotation by Keras.



CONCLUSION

- Learning latent features of images through dimensional reduction allows auto-encoder to do a good job of denoising the images.

- Generation of images through Keras causes for stretching of the corners to preserve dimensions, this introduces "fake" structures that trick the network.



Future research

- Expanding to the original 1200X1200 images, this has the potential of improving the network since the cells are mostly centered, so image generation will not trick it as much.
- Modify the image generated to add zeroes at the corner instead of stretching to preserve the dimensions.

Resources

- Dertat, Arden, and Arden Dertat. "Applied Deep Learning - Part 3: Autoencoders." *Towards Data Science*, Towards Data Science, 3 Oct. 2017, towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798.
- Rohrer, Brandom. *"How Convolutional Neural Networks Work." YouTube, 18 Aug. 2016.*
- *The code is written by the help of the course worksheets and keras documentation.*

Acknowledgements

I'd like to thank Prof. Hughes and Prof. Kural for guidance and patience throughout this semester and for supervising this project.
Also Scott, Nathan and Umida for help with the experimental part of obtaining the data.
Finally, my sister for flying from Cairo to be with me even when I am busy.

THE OHIO STATE UNIVERSITY

Final Project
Spring 2019
PHYS6820