Society for Artificial Intelligence and Deep Learning

SAiDL

Summer 2020 Induction Assignment

August 2020

1. About

This is the official Summer assignment for SAiDL inductions. Please fill this Google Form to register and join our Slack channel. At the end of the assignment, please fill the submission form. If you have any doubts, try to Google it, if still not clear you can pm any SAiDL member. The deadline for the assignment is **24th August 2020**

2. Getting Started

2.1. Preparing your Machine

- Dual boot your machine with Ubuntu 18.04LTS
- Install git
- Install the latest version of Anaconda with Python 3.6 (make sure that you agree to change PATH variable when prompted)
- Install OpenCV via python for pip
- Install the latest version of Pytorch with or without GPU support, depending on your system specs.

2.2. Setting up your Github account

Set up a GitHub account and make sure that you redeem the Student Developer plan here.

2.3. General Notes

Before you try and attempt the assignment, it is important that you remember we value your attempt more than the final results. You may or may not have completed the assignment as a whole, but make sure to submit it.

3. Recommended Study Resources

3.1. Python 3.6

Python is the Franca Lingua of Ai. You will be learning Python 3.6. Do it either from "Learn Python 3 the Hard Way by Zed Shaw" or tutorial series by Sentdex

3.2. Coursera Course on Machine Learning

This course is the launchpad for most Ai enthusiasts and gives you your first hands-on approach to Machine Learning along with the maths behind it. You will be implementing ML algorithms in Octave/MATLAB.

3.3. Linux Terminal

This is one of the fundamental requirements for any computer scientist. This will help you get started.

3.4. Numpy

Crudely speaking this is MATLAB for Python. We suggest you do this after the Andrew NG course. You can go to its official tutorial or learn it with hands-on deep learning experience via deeplearning.ai's first course.

3.5. Pandas

This is one of the most crucial and powerful libraries in data science. To begin with, you end up learning how to read and write CSV and JSON files as well as how to manipulate Data Frame rows, columns, and contents. Again Sentdex to the rescue.

3.6. Matplotlib

Learn how to plot basic graphs. The pyplot submodule should be enough for the beginning. Sentdex is your savior again.

3.7. OpenCV

OpenCV is one of the best computer vision libraries out there, However, currently, we will be using it only to load images as Numpy arrays. Tutorial for the same is here.

3.8. Stanford's Courses for NLP, CV, RL

You have to see all the lecture videos of Stanford's Classes for Computer Vision, Reinforcement Learning, or Natural Language Processing using Deep Learning. If possible, do the course assignments too.

Natural Language Processing CS224: [Youtube link] [Course Link]
Computer Vision CS231: [Youtube link] [Course Link]
Reinforcement Learning CS234: [Youtube link] [Course Link]

3.9. Gym

OpenAI's gym is a toolkit for the development of reinforcement learning algorithms. It has many agents and environments for which state, action, observation, returns based on actions, etc. can be generated. You can follow the official documentation.

3.10. Pytorch Tutorials

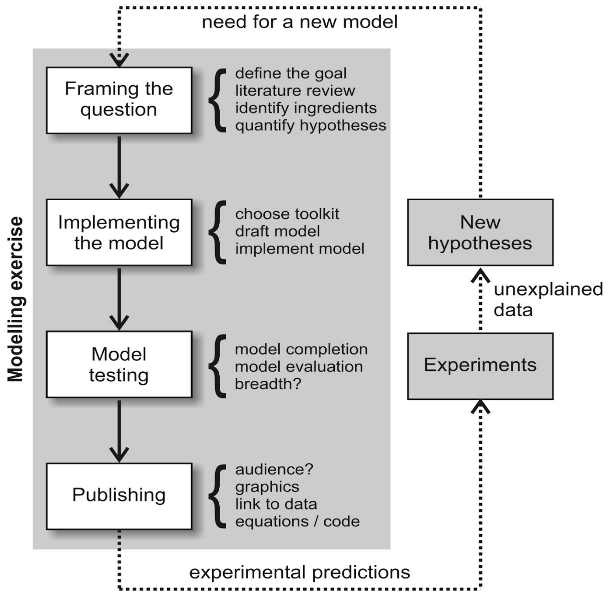
You will be doing the assignments in Pytorch and Numpy only. We would suggest that you get hands-on experience with Pytorch by following the official tutorials or this series.

4. Assignment:

Question 4.1 is compulsory. Additionally, you can **EITHER** attempt two questions out of Section 4.2 **OR** attempt Paper Implementation as Question 4.3 Please note that the difficulty for these questions does vary, and we will consider the effort you have put and the number of things you tried out.

4.1. Research Problem Formulation and Critical Analysis (Compulsory)

A recent paper by Gunnar Blohm, Konrad P. Kording, and Paul R. Schrater, titled "A How-to-Model Guide for Neuroscience" (Some notes for this paper), breaks down the process of doing research into 10 practical steps. Although this paper is for Neuroscience, it applies to other fields as well because the steps are not neuroscience specific.



Your task is to **break down any new project idea/existing project into these steps** (if needed, you can skip the last step, i.e., publishing). **No code is required** (if needed, you can just assume that the code was implemented for steps 7 and 8), only a plan for a project broken down into these steps; however, if you are writing about an existing project, you can provide a link to the code. The project can be anything as long as it is related to Artificial intelligence. Please assume no constraints on the dataset and computational power, focus on the questions and hypothesis, not on the datasets or feasibility. **There is no word limit, but we would recommend keeping it short and to the point. We want to instill critical thinking. Hence the task is very subjective and has no right or wrong answers.**

If you don't have an existing idea/project you can also take inspiration/build upon/use your knowledge from any one of the other questions in the assignment.

Please go through the paper for an example. Also, note that any new idea that you come up with belongs to you, **we do not store/use/take any credit for it**.

4.2. Tasks (Attempt **either** two tasks under this or 4.2)

4.2.1 Computer Vision

The question is based on computer vision, specifically classification tasks and generative adversarial networks.

Tasks (following the given order will help):

1. Use the LeNet architecture, and train a classifier on the **CIFAR10 dataset** (Hyperparameter values are left to you to experiment with and tune). Note the final test accuracy you obtained.
2. Train a DCGAN on the CIFAR10 dataset, and use the images generated from it to ***augment*** your dataset. Train the classifier on this augmented dataset and note the test accuracy once again. Did the accuracy increase or decrease substantially, or did it remain relatively the same? Did the result agree or disagree with what you expected?
3. We now extend this study on augmentation to a broader variety of augmentation techniques. Experiment with other augmentation techniques such as (but not restricted to) random crops, image flipping, and adding noise. Log your results and compare them with each other. Try to come up with explanations for why one might have worked better than the other.

Bonus:

1. Try combining various augmentation techniques and check if it improves your results.
2. Repeat the above study on augmentation techniques with the MNIST dataset. Does using them give you different results from when you applied them on CIFAR10 i.e. is the effectiveness of data augmentation techniques dependent on the data itself? Once again, log your results and compare them with the results from CIFAR10. Furthermore, note the differences, and reason about why these differences occurred.
3. Train a classifier on MNIST. Now, use image flipping to augment MNIST and train your classifier once again. Do you notice an increase or decrease in performance? Is it different from what you observed when doing the same with CIFAR10? Give possible reasons for the same.

References:

1. LeNet architecture
2. DCGAN

4.2.2 Natural Language Processing

The question is based on natural language processing, specifically, sequence classification or text classification. Text classification is a standard problem in NLP.
In this task, you will be given a dataset consisting of Reddit posts consisting of their content, title, author, and **flair** taken from a particular subreddit. You have to build an automated system that considers the provided data to detect the flair that should be assigned to a specific post, and it is essentially a multi-class classification task. We will consider F1 Scores [1] (micro average and macro average) for the evaluation.
Link to the dataset: **Repository**

Tasks (In Order):

1. Design a Recurrent Neural Network (RNN / LSTM / GRU) based model utilizing Word Embeddings [2] to build a strong baseline.
2. Add an attention layer on top of LSTM outputs, and compare it with the baseline in Task 1. It is already implemented here.
3. Visualize the attention weights for some samples and write down a qualitative analysis.

Important guidelines:

1. Weak baseline performance for this question is: **[Micro F1 = 0.50, Macro F1=0.17]** you must beat this baseline performance.
2. The code will not be accepted without proper documentation and comments and mention your performance in the documentation.
3. One should try to get the strongest baseline possible in Task-1
    a. There are various hyperparameters in RNN/LSTM/GRU layers, like hidden size, number of layers, dropout, directionality.
    b. Moreover, there can be various ways to take the output from LSTM, last step, pooling, etc.
4. You will be given training and development datasets for analyzing your results however final evaluation will be done on a test set that will be made public at the end of the task.

Bonus:
Improve the given model by making changes to the implementation in Task-1 and Task-2. You are free to use any techniques for improving your scores however usage of external datasets is not allowed.

References:

1. F1 Score
2. Introductory Blog to Word Embeddings
3. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." ICLR 2015.

4.2.3 Reinforcement Learning

Reinforcement Learning is a very interesting and unique field of AI. In this assignment, we will tackle a pretty cool application of RL, which is to play games. More specifically, you will be training an RL agent to finally learn how to compete in a Pokemon Battle!

There wasn't any readymade environment you could directly start working with, so we made a small codebase which you can build upon. We have defined a few guidelines and written a bunch of starter codes so you can concentrate more on building an efficient agent. You can find the repo at - PokeBattle

Problem Statement:

It is the year 2018. Ash Ketchum has still not won any Pokemon League yet. He realizes he cannot be good enough to win by himself but he has still not left all hope. He approaches Professor Oak for advice. you are a good friend of Oak, and he knows that you are interested in Reinforcement Learning. Since Ash is not good enough yet and you don't exist in the Pokemon world, he realizes you both can help each other. Your job is to design an RL agent that can learn how to play a Pokemon Battle and help Ash Ketchum win his first trophy!

Guidelines:

Go through the Github repo and find out what code to implement and how to go about it. The parameters you will be judged upon are:

- Completion of starter code (there are a lot of things that are still not entirely implemented in the environment).
- Implementation of additional features (like adding more moves, more status effects, more complexity in other areas). Take a look at the repo (and the issues page) for more ideas.
- Neat and clear code with docstrings and comments to explain the choice of Agent class methods and operations.
- Explain what your agent is achieving and clearly present what your agent is learning.
- How well you have defined the reward function and state space and how relevant they are in helping your agent learn.

Resources:

1. Barto and Sutton, "Reinforcement Learning: An Introduction"
2. David Silver or Emma Brunskill Reinforcement Learning courses on YouTube
3. A (Long) Peek into Reinforcement Learning
4. GenRL: A PyTorch reinforcement learning library for generalizable and reproducible algorithm implementations

4.3. Paper Implementation

If you are attempting this question your task is to implement an in-depth learning paper. Select any one of the papers published in top conferences (**NeurIPS, ICML, ICLR, ACL,** etc). First, read the abstract and give a rough reading of the paper you have chosen. **At this point inform us that you are attempting to implement the paper and tell us about the paper chosen**. Once you get approval from us, you can go ahead and give a thorough reading of the paper and try implementing the paper using pytorch.

We are not expecting you to write a very clean implementation but will be testing you on your understanding of the paper. Your code should be able to replicate the results described in the paper. If possible, add hyperlinks to accepted papers and link from Papers with Code

Document your code with a readme and a review for the paper (preferably in Latex). The paper review might contain limitations of the paper, could the authors have done some changes to get better results and some new insights that you might have for the paper. Remember this task has not been designed for beginners but anyone is welcome to give it a try.

5. Submission

After completing the assignment (or paper implementation), upload your code on Github in a repository titled SAiDL-Summer-Assignment-2020. The **Deadline is 24th August 2020**. You can learn how to use git from codeacademy.

You are **not** allowed to collaborate with others. Any form of plagiarism will be penalized. In case of doubts, please find the contact details of the mentors below. To submit, fill this submission form. Inform us when you complete each question from the assignment (submit the google form only after completion of the whole assignment or after deadline). Maintain all the code that you write in **one private repository** (you have to submit this in the form given above. Make sure the repository is private before the assignment's deadline, after which you can make it public). For whichever question you attempt, **write your approach in Word/Docs/Latex form, and convert it to a PDF**. Join our Slack channel for receiving all updates and all queries.

NOTE - SAiDL holds the right to the questions used in this assignment, if you choose to use this outside of the assignment, you will have to cite/acknowledge SAiDL.

6. Contact Info

In case of any doubts, clarifications or guidance needed, you can contact one of us:

- Aditya Ahuja (President) - 9834530623
- Het Shah (Vice President) - 7984336580
- Alish Dipani (Q 4.1) - 80555 65051
- Gaurav Iyer (CV) - 9819451499
- Somesh Singh (NLP) - 9140827410
- Sampreet Arthi (RL) - 9359788418
- Shashank M (NLP) - 9986048012