Neural Lego: A Code-Free Neural Network Editor for Deep Learning Enthusiasts
Proposal for COMP208

Team Numbers of Group 35:
Qirui Bao, Pin Qian, Ruijie Ye, Haomiao Ji, Haonan Wang, Shujie Yang

1. Project Description

As artificial intelligence advances, a rising number of people from various industries and research fields are keen to master deep learning (DL). Our project is tailored for all those AI enthusiasts, whether novices or specialists, allowing them to develop AI models without writing any code. Concretely, there are two key motives and features of our project. First, considering that constructing an AI model comes with a lot of repeatability and modularity, such as data preprocessing, network construction and result visualization, we facilitate users to build models by drawing a flowchart of the network topology via graphical interfaces instead of writing pythonic code. Next, we provide a modular AI programming community based on the core feature mentioned above, forming a technical discussion environment. In brief, our project is a web-based platform that assists users to better utilize and explore DL.

2. Aims & Objectives

2.1 Aims

We expect to achieve three aims upon completion of this project.

- **Lowering the learning threshold for DL**
  For DL entry-level users, we aim that by using the functionality of our graphical interface to build neural networks, they would find it easier to comprehend and get started in the subject of deep learning.
- **Saving programming time**
  For DL high-level users, we aim that our platform can enable them to save time writing code so they can focus more on research work or industrial demands.
- **Providing a modular AI programming community**
  For all users, we aim that a modular AI programming community could be formed to bring AI enthusiasts together, allowing for technical networking.

2.2 Objectives

The objective of our project is three-fold.

- **User Information Management**
  - Realize the authentication module including register and login
- **Graphical interface functionality for network building**
  - Realize the functionality of flowchart creation and edition
  - Achieve the functionality of transforming flowchart to code
  - Implement code-free training processes from data preprocessing to result visualization
- **Community functionality for user interaction**
  - Realize the functionality of project sharing via template gallery
  - Realize the networking functionality through discussion board
  - Implement the functionality of customized recommendation of projects

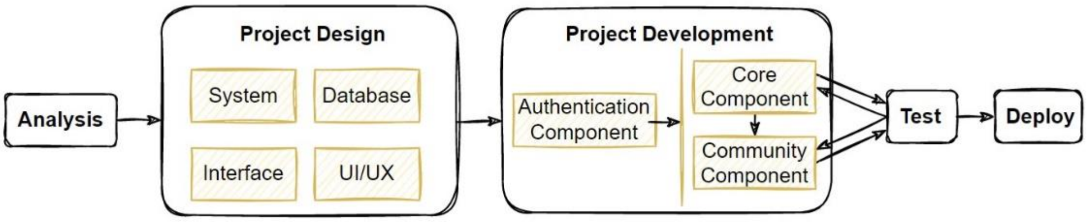
3. Key Literature & Background Reading

There is an evolutionary of framework for DL. MATLAB and OpenNN were used initially, however, their APIs were complex [1]. After that, some DL frameworks such as Torch, Caffe, Chainer, and Theano emerged and laid the basis for later developments [2]. Later, TensorFlow was released which draw on Theano's declarative style, and PyTorch inherits Torch's imperative style [2]. Despite recent improvements to popular frameworks such as PyTorch and TensorFlow, users still face a barrier to entry, and the issue of repetition code in AI model creation remains unsolved.

In addressing these problems, some efforts have been made. Hugging Face [4] integrates a large number of existing models and databases for AI learners. However, using it still requires writing a certain amount of code. Scratch [3] is a graphical programming software for young children. For tiny programs, it can be implemented without writing code, but for DL projects, it cannot achieve such needs.

Based on the foregoing, we believe there is a need to develop our proposed project, since it not only makes it easier and faster for people to learn deep learning, but also better improve the existing framework.

4. Development & Implementation Summary



In this section, our general workflow is presented. As the figure shows, the process of our project is organized into five stages: requirements analysis, design, development, testing, and deployment.

To fulfil aims and objectives, the development stage is divided into three components. The authentication module is initially set up to handle logging in, registering, and verifying that whether the user has access to other components. The code component will then be implemented in order to accomplish the capability of producing flowcharts by users and converting them to code by our server. In addition, the community component will be realized based on the above parts. Our project is test-driven and Agile development model id leveraged. Thus, the completion of each module covers both development and testing. Detailed testing procedures is contained in Section 6.

For implementation, we employ a front-end and back-end separation for each component. The team members are divided into three groups. At each stage of development, the test engineer group first complete the test code according to the test documentation. The front-end engineer group realize the front-end page and page logic according to the UI/UX design, meanwhile, the back-end engineer group implemented the functions according to the API documentation. After integrating the front and back ends, the code is then handed over to the test engineer group for code testing until the test was completed.

The following table depict our development environment, implementation languages, libraries that may be used, and the rationale for choosing them.

IDE	VS code	Most team members are familiar with VS code, which is lightweight and rich in plugins for convenient development.
System Environment	ubuntu 20.04	Flexible and stable, with easy environment configuration.
Front end	H5, CSS, JS	Basic web development language. We may use vue.js for page beautification.
	ECharts.js	We use this library to visualize training result.
Back end	Python 3.9.0	Compared to Java Spring Boot, Django is much lighter and has enough features to achieve our needs. Compared to Flask, Django has a more mature ecology makes the basic web development process easier and allows us to devote more time to the implementation of core features.
	Django 4.0.0	
Database	MySQL	In our project, the data stored is more suitable for a relational database. The team members are familiar with MySQL, so it is chosen.

5. Data Sources

In our project, there are three phases involving data sources.

5.1 Project Testing Phase

We will use the Iris dataset during the testing phase to ensure that the project is working properly. The Iris dataset is a classic dataset widely used as a classification task for beginners. We download it from Kaggle [5] with permission.

5.2 Project Evaluation Phase

At this stage, the evaluation questionnaire will generate participant data such as questionnaire results, personal details, etc. we will anonymize these data, store them on our university file store and delete them at the end of the academic year. We will not collect, use participants' data and collect information without participants' permission.

5.3 User Usage Phase

At this phase, Users can upload their own datasets and customize whether or not to make their datasets public or private. Our platform obtains the user's selection and permission by asking, ticking and checking and ensures that personal datasets are anonymous to ensure the confidentiality. Furthermore, the administrator of our platform reviews the uploaded data to ensure that it complies with the terms of the license, policies and copyright notices.

6. Testing & Evaluation

6.1 Testing

We plan to adopt Agile as our development model and make our project test-driven, which means that the test case should be written before development. Bottom-up unit tests will be conducted to perform functional testing on the three components (described in Section 4.1) to ensure that each functional module is working properly. After that, module test is required to ensure the coherence of the completed component.

6.2 Evaluation

We intend to invite volunteers as beta testers to evaluate our website when testing-driven development is finished. The contents to be evaluated are divided into two aspects: engineer evaluation and user evaluation.

- **Engineer Evaluation**
  The test engineer will conduct a detailed evaluation on the usability, stress testing and other non-functional tests of our project according to user usage.
- **User Evaluation**
  We intend to use evaluation questionnaires to receive feedback from volunteers after using our platform, so as to further improve and optimize the UI design and technical implementation of our project.

7. Ethical Considerations

We have read the Ethical Guidelines and the Policy on Research Ethics carefully and will follow them.

7.1 Data Source

It can refer to the content in Section 5.1 and 5.3.

7.2 Testing Software with Human Participants

The evaluation we performed can be referenced from Section 6.2. Before evaluating the project, we invite participants who give informed consent and explain our program to them without forcing anyone to participate or quit. With their consent, participants are able to test our platform without any sensitive information. If they drop out we will remove the data collected from them.

7.3 Protecting Participant Data

It can refer to the content in Section 5.2.

8. BCS Project Criteria

8.1 Practical and Analytical Skills

For practical skills acquisition, we are going to complete the development of projects through self-directed learning of the skills required. For analytical skills acquisition, we analyzed the shortcomings of the existing framework by information search (see section 3) and analyzed the requirements of the project.

8.2 Innovation and/or Creativity

We came up with the innovative idea of using graphical interfaces to build neural network without writing codes, which has not been done before.

8.3 Synthesis Capability and Quality Solution

We put forward an implementation and evaluation plan based on the innovative ideas and information collected online and learned from COMP201 (see section 4).

8.4 Real Need in a Wider Context

Our project meets the needs of DL enthusiasts at all levels and has advantages that are missing from existing frameworks (see section 2,3).

8.5 An ability to self-manage a significant piece of work

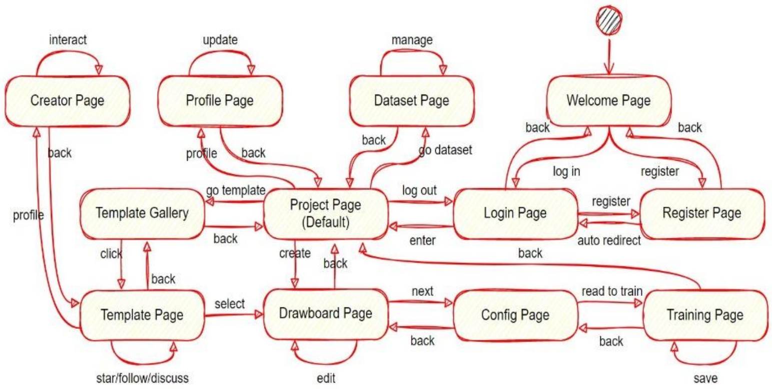
Our group members meet every week to report on the work they have done during the week, and clarify what they will do in the next week.

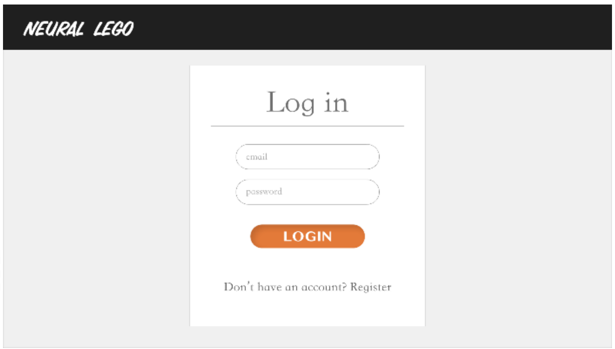
8.6 Critical Self-Evaluation of the Process

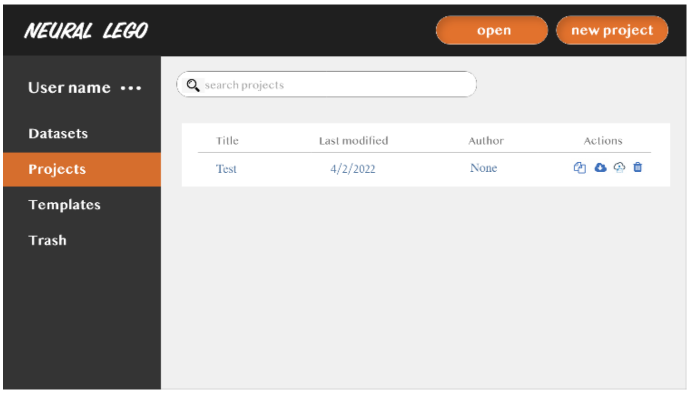
We summarize and reflect on the shortcomings of last week and discuss how to make the project more refined at the weekly meeting.

9. UI/UX Mockup

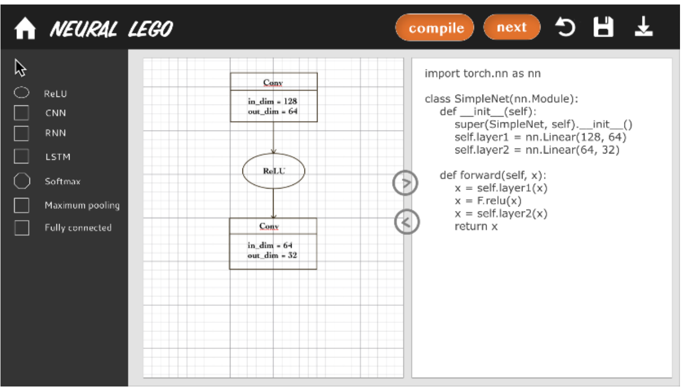
Due to the limited space, we give out the UX state transition diagram here, and some UI design of the webpages.







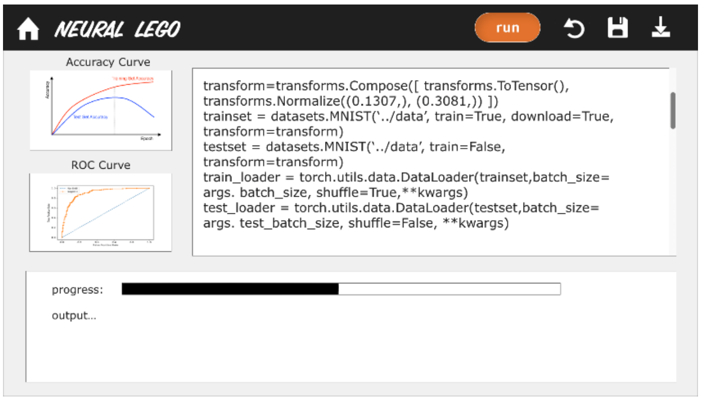
Title	Last modified	Author	Actions
Test	4/2/2022	None	



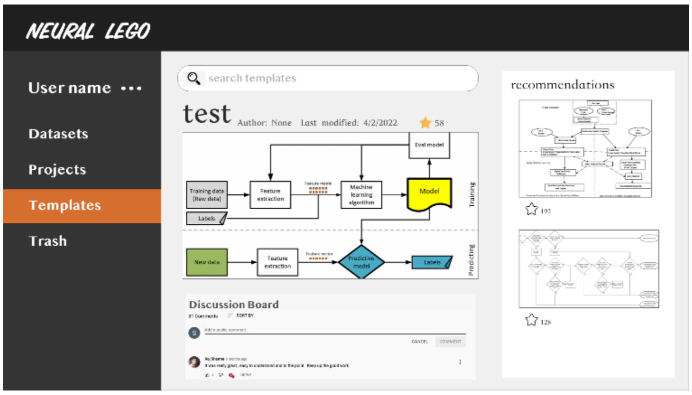
```
import torch.nn as nn

class SimpleNet(nn.Module):
    def __init__(self):
        super(SimpleNet, self).__init__()
        self.layer1 = nn.Linear(128, 64)
        self.layer2 = nn.Linear(64, 32)

    def forward(self, x):
        x = self.layer1(x)
        x = F.relu(x)
        x = self.layer2(x)
        return x
```

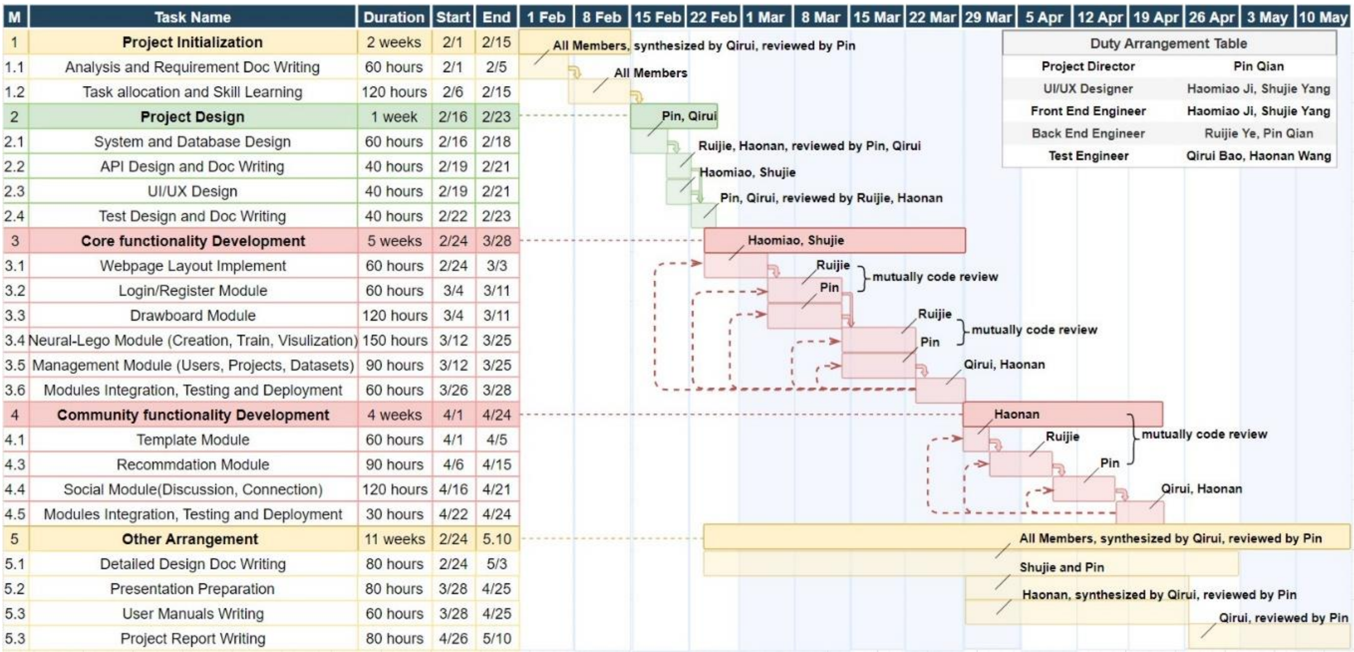


```
transform=transforms.Compose([ transforms.ToTensor(),
transforms.Normalize((0.1307,), (0.3081,)) ])
trainset = datasets.MNIST('../data', train=True, download=True,
transform=transform)
testset = datasets.MNIST('../data', train=False,
transform=transform)
train_loader = torch.utils.data.DataLoader(trainset,batch_size=
args. batch_size, shuffle=True,**kwargs)
test_loader = torch.utils.data.DataLoader(testset,batch_size=
args. test_batch_size, shuffle=False, **kwargs)
```



10. Project Plan including Risks & Contingency Plans

10.1 Project Plan and Duty Arrangement



M	Task Name	Duration	Start	End
1	**Project Initialization**	2 weeks	2/1	2/15
1.1	Analysis and Requirement Doc Writing	60 hours	2/1	2/5
1.2	Task allocation and Skill Learning	120 hours	2/6	2/15
2	**Project Design**	1 week	2/16	2/23
2.1	System and Database Design	60 hours	2/16	2/18
2.2	API Design and Doc Writing	40 hours	2/19	2/21
2.3	UI/UX Design	40 hours	2/19	2/21
2.4	Test Design and Doc Writing	40 hours	2/22	2/23
3	**Core functionality Development**	5 weeks	2/24	3/28
3.1	Webpage Layout Implement	60 hours	2/24	3/3
3.2	Login/Register Module	60 hours	3/4	3/11
3.3	Drawboard Module	120 hours	3/4	3/11
3.4	Neural-Lego Module (Creation, Train, Visulization)	150 hours	3/12	3/25
3.5	Management Module (Users, Projects, Datasets)	90 hours	3/12	3/25
3.6	Modules Integration, Testing and Deployment	60 hours	3/26	3/28
4	**Community functionality Development**	4 weeks	4/1	4/24
4.1	Template Module	60 hours	4/1	4/5
4.3	Recommdation Module	90 hours	4/6	4/15
4.4	Social Module(Discussion, Connection)	120 hours	4/16	4/21
4.5	Modules Integration, Testing and Deployment	30 hours	4/22	4/24
5	**Other Arrangement**	11 weeks	2/24	5.10
5.1	Detailed Design Doc Writing	80 hours	2/24	5/3
5.2	Presentation Preparation	80 hours	3/28	4/25
5.3	User Manuals Writing	60 hours	3/28	4/25
5.3	Project Report Writing	80 hours	4/26	5/10

Duty Arrangement Table	
Project Director	Pin Qian
UI/UX Designer	Haomiao Ji, Shujie Yang
Front End Engineer	Haomiao Ji, Shujie Yang
Back End Engineer	Ruijie Ye, Pin Qian
Test Engineer	Qirui Bao, Haonan Wang

10.2 Risks and Contingency Plans

Risk	Contingency	Likelihood	Impact
The time required to develop the software is insufficient.	First develop the essential function so that the software could operate even if optional functions are not completed.	High	Serious. The final project could not operate as expected.
Key members are ill or not able to participate at critical points for some reasons.	Assign each module (or significant modules) to more than two persons so others can take over someone's job.	Moderate	Serious. There could be a missing part in the software.
Task fails to be finished as scheduled	Make the schedule allows for a week cushion.	High	Serious. This can lead to incomplete projects.
Different modules cannot be integrated as expected.	Redesign or reproduce the interface to adapt to these modules.	Moderate	Catastrophic, it leads to a failure of the project.
Invisible bugs in the software occur.	Strictly follow the black-box and white-box tests to avoid bugs	High	Serious. Bugs may decrease user experience.

11. References

1. Chung, Y., Ahn, S., Yang, J. and Lee, J., 2017. Comparison of deep learning frameworks: about theano, tensorflow, and cognitive toolkit. *Journal of Intelligence and Information Systems*, *23*(2), pp.1-17.
2. Yuan, L., 2020. A Brief History of Deep Learning Frameworks. [Blog] *apeforest*, Available at: <https://towardsdatascience.com/a-brief-history-of-deep-learning-frameworks-8debf3ba6607> [Accessed 13 February 2022].
3. Maloney, J., Resnick, M., Rusk, N., Silverman, B. and Eastmond, E., 2010. The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, *10*(4), pp.1-15.
4. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M. and Davison, J., 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.
5. Kaggle.com. 2017. *Iris Species*. [online] Available at: <https://www.kaggle.com/uciml/iris/activity> [Accessed 15 February 2022].