NeuroBDSM- Boss, Detractor, Solver and Master in role-playing games of neural networks for search of a solution

“The right solution is always beautiful. But every problem always has a beautiful wrong solution.”

“Sometimes there is no right solution, but there is always a better solution.”

Lyubimkov Dmitri

E-mail: loftlong@gmail.com

2025

Abstract

This article introduces **NeuroBDSM**, a novel idea for collaborative problem-solving using multiple neural networks, each assigned a distinct role: **Boss**, **Detractor**, **Solver**, and **Master**. The framework aims to address the limitations of traditional human-AI collaboration, where human oversight and expertise are required throughout the solution-finding process. By replacing the human with a team of specialized neural networks, NeuroBDSM enables autonomous, iterative problem-solving. The **Solver** generates solutions, the **Detractor** critiques them, the **Master** provides expert guidance, and the **Boss** oversees the process, ensuring optimal results and terminating the search when a satisfactory solution is reached. The system also allows for optional human intervention, combining the benefits of autonomous AI with human expertise. A practical implementation using the OpenAI API is provided, demonstrating the framework's adaptability to various tasks and models. NeuroBDSM represents a significant step toward fully autonomous AI systems capable of complex, multi-role collaboration, with potential applications in diverse fields requiring iterative problem-solving and critical analysis.

Table of contents

1. Search for a solution in a pair of people and a neural network

What does the process of finding a solution to a task or problem look like now in a pair of people + ChatGPT (or any other neural network with chat capabilities). A person sets a task or formulates a problem. The neural network offers solutions. The user asks clarifying questions or criticizes the proposed solution or asks to clarify something unclear. ChatGPT responds in the next iteration and the process of finding a solution continues until it is found or a dead-end situation is created. This process has several disadvantages, both obvious and hidden. It requires the presence of a human user throughout the entire process of finding a solution and requires the user to have sufficient competence in the area under discussion to see errors and shortcomings. The neural network itself, being aimed at fulfilling the user's requirements and wishes, strives to give an answer ignoring its shortcomings.

2. Finding a solution without a human

Let's exclude a person from the search for a solution. Let's replace him with another neural network, or better yet, several neural networks. Let's assign each neural network its own role and call them one by one, showing each the results of all previous dialogues. In addition to the problem/task itself, we'll add a description of the solution search process.

There are 4 independent neural networks, each performing its own role. Their joint task is to jointly generate the correct answers to user requests. Neural networks have 4 roles: <Solver>, <Detractor>, <Master>, <Boss>. Neural networks are called in turn in the following order: <Solver>, <Detractor>, <Master>, <Boss>. The <Solver> role generates answers to the questions and tasks posed. <Solver> must review <Detractor>'s criticism, <Master>'s suggestions, and <Boss>'s directions from previous iterations. In each iteration, <Solver> must develop a solution based on a full dialogue between all participants. The <Detractor> role criticizes the latest response of the <Solver> neural network, but does not offer any solutions of its own. The <Master> role is an expert who analyzes only the latest <Solver> answers and <Detractor> criticism, evaluates them, and suggests further directions for reflection. The <Master> role does not offer its own solutions or criticize. Choosing the right answers is the <Master> role's job. The <Boss> role provides overall process management, but does not interfere with the technical part of the process. <Boss> does not propose its own solutions, does not criticize the proposed solutions, and does not evaluate solutions from a technical point of view. Sometimes there is no right solution, but there is always a better solution. Finding the best solutions is the job of the <Boss> role. <Boss> can stop generating an answer if it believes that the optimal result has already been achieved. To do this, it must include the word "Fluggaenkoechiebolsen" in its answer. But the user can continue solving if he is not satisfied with the last answer of <Solver>.

If after the end of one iteration a decision was made to search for a better answer, then the sequence <Solver> -> <Detractor> -> <Master> -> <Boss> is called again.

3. Finding a solution without humans with the possibility of intervention

If after each iteration <Solver> -> <Detractor> -> <Master> -> <Boss> we add the ability for the user to write his own review, then we can combine the first and second approaches. If the user believes that the process is going in the right direction, then he (or the program) can write the word “Continue”.

4. Program

The program requires the openai library to work. You can install it using the command

pip install openai

The program supports any openai-compatible API (OpenAI itself, DeepSeek, ...) To configure the API you need, fill in the keys in the unpwdkeys.py file and change the model and base_url fields of the llms variable in the cfg.py file. You can use different models for different roles and even different APIs for different roles.

The program is launched by running the file neurobdsm.py

The program executes tasks from the task variable in the cfg.py file. Currently, the variable contains questions on previous articles: https://github.com/loftyara/LLMagogy and https://github.com/loftyara/CLLM_void_zones

5. Files

5.1. File unpwdkeys.py

Add here api keys for neural networks of each role.

5.2. File cfg.py

This file contains various parameters/variables:

- proxy – fill in the variable if access to openai compatible API occurs through a proxy server
- roles – description of each role
- llms – LLM fulfilling each role. Different LLMs and different models can be used
- tasks – tasks, questions that LLMs must answer. Insert your tasks here
- min_iterations – minimum number of iterations <Solver> -> <Detractor> -> <Master> -> <Boss>
- max_iterations – maximum iterations <Solver> -> <Detractor> -> <Master> -> <Boss>
- stop_word – word for <Boss> role to complete the search for an answer

5.3. File neurobdsm.py

The main program file. Will execute each task from cfg.tasks one by one. If you want to add a user reaction, replace the lines

```
for rid, message in messages.items():
    message.append({'role':'user', 'content':[{'type':'text', 'text':'Continue'}]})
```

to enter a command from the user with its addition to the list of messages