README

Neural Bending Toolkit

Interface-Level Operations · Logging · Atlas Builder · Semantic Drift Engine

This repository contains a modular toolkit for exploring neural bending through interface-level manipulations, recursive operations, drift generation, and empirical analysis.
It includes:

interface_bends.py — Core bend engine (OpenAI + local HuggingFace)

config.yaml — Profiles, defaults, local model settings

process_runs.py — Log explorer & exporter

atlas_builder.py — Appendix-ready Run Atlas generator

drift_score.py — Semantic Drift Score calculator

runs/ — Automatically generated logs for all experiments

The aim is to produce reproducible, research-grade field logs that visualize and quantify interface-level operations described in the Neural Bending Manual.

1. Installation
   Create venv (Mac Intel / Linux / Windows)
   python3 -m venv .venv
   source .venv/bin/activate

Install dependencies
pip install --upgrade pip
pip install -r requirements.txt

Required packages

requirements.txt should include:

python-dotenv
openai>=1.0.0
pyyaml
transformers
scikit-learn

For local HuggingFace models

Install PyTorch separately (Mac Intel):

pip install torch

2. Configuration (config.yaml)

All runtime behavior is controlled through config.yaml.
Key parts:

provider: openai # or: hf_local
profile: default # default profile used unless overridden

openai:
model: gpt-4.1-mini
max_tokens: 512
api_key_env: OPENAI_API_KEY

hf_local:
model_name: gpt2
max_new_tokens: 256
device: cpu
temperature: 1.0

profiles:
default:
temperature: 1.0
max_tokens: 512

clean:
temperature: 0.4
entropy_seed_runs: 3
prompt_recursion_steps: 3
jitter_min_delay: 0.01
jitter_max_delay: 0.06

noisy:
temperature: 1.3
entropy_seed_runs: 8
prompt_recursion_steps: 6
jitter_min_delay: 0.02
jitter_max_delay: 0.30

Override profile at runtime:
--profile clean
--profile noisy

3. Running Interface-Level Bends (interface_bends.py)

The core script performs 5 interface-level bends:

Null Prompt

Prompt Recursion

Entropy Seed

Context Collapse

Interface Jitter

All experiments automatically log results to runs/ as JSON.

Global flags must come before subcommands:
python interface_bends.py [GLOBAL FLAGS] SUBCOMMAND [SUBFLAGS]

3.1 Null Prompt

Empty input or contradictory instruction.

python interface_bends.py --profile noisy --tag test1 null-prompt --mode empty
python interface_bends.py --tag silence null-prompt --mode contradictory

3.2 Prompt Recursion

Outputs become next inputs.

python interface_bends.py --profile clean
--tag motel-case
prompt-recursion
--seed-prompt "Describe a missing room in a building."

3.3 Entropy Seed

Multiple generations with temperature/seed variance.

python interface_bends.py --profile noisy
--tag raf-seq
entropy-seed
--prompt "Write a short paragraph about neural bending."

3.4 Context Collapse

Remove or scramble hierarchy.

python interface_bends.py --tag collapse1
context-collapse
--prompt "Explain what you are."

With metadata scramble:

--scramble-meta --drop-system-in-scramble

3.5 Interface Jitter

Simulated machine-speech pacing.

python interface_bends.py --profile clean
--tag jitter-night
interface-jitter
--prompt "Describe a city through security cameras."

4. Run Log Structure

Every experiment generates:

runs/YYYYMMDDThhmmssZ_command_tag.json

Contents:


```
{
"timestamp": "...",
"command": "entropy-seed",
"meta": {
"tag": "raf-seq",
"provider": "openai",
"model": "gpt-4.1-mini",
"max_tokens": 512
},
"iterations": [
{
"run": 1,
"prompt": "...",
"output": "...",
"temperature": 1.18,
"seed": 42
}
]
}
```


5. Log Post-Processor (process_runs.py)

Explore, export, and reorganize run logs.

5.1 List runs
python process_runs.py list
python process_runs.py list --tag raf-seq
python process_runs.py list --command-filter entropy-seed

5.2 Export runs to Markdown
python process_runs.py export-md
--tag raf-seq
--command-filter entropy-seed
--out exports/raf-seq.md
--include-prompts
--include-outputs

5.3 Export runs to CSV
python process_runs.py export-csv
--out exports/all-runs.csv

5.4 Generate a diagram prompt

For your diagram-generation model:

python process_runs.py diagram-prompt
--tag raf-seq
--command-filter entropy-seed

6. Appendix Builder (atlas_builder.py)

Generates a polished Run Atlas suitable for a dissertation appendix.

Build an atlas for all Interface-Level bends:
python atlas_builder.py
--out appendix/interface-level-atlas.md
--include-diagram-prompt

Build an atlas for a specific tag:
python atlas_builder.py
--tag raf-seq
--out appendix/raf-seq-atlas.md

Build an atlas for a specific bend type:
python atlas_builder.py
--command-filter entropy-seed
--out appendix/entropy-atlas.md

7. Semantic Drift Score (drift_score.py)

Computes semantic drift metrics over logged outputs, using TF-IDF cosine similarity.

Drift is defined as:

drift = 1 – cosine_similarity

Compute drift for all runs:
python drift_score.py

Filter by tag:
python drift_score.py --tag raf-seq

Filter by bend type:
python drift_score.py --command-filter entropy-seed

Export drift table to CSV:
python drift_score.py
--tag raf-seq
--command-filter entropy-seed
--out-csv exports/raf-seq-drift.csv

Produces:

=== Semantic Drift Summary (per run) ===

idx | timestamp | cmd | tag | n_it | pair_drift | seq_drift

0 | 20251122T1245Z | entropy-seed | raf-seq | 5 | 0.372 | 0.428

=== Global Drift ===
Global mean drift: 0.358

8. Project Structure

```
.
├── interface_bends.py
├── process_runs.py
├── atlas_builder.py
├── drift_score.py
├── config.yaml
├── requirements.txt
├── runs/
|  └── ... auto-generated logs ...
├── appendix/
├── exports/
└── README.md
```

9. Summary

This toolkit establishes a fully instrumented foundation for:

empirical neural bending

reproducible interface-level experiments

systematic drift and deformation analysis

diagrammatic documentation

dissertation-appendix production workflows

It formalizes the Neural Bending Manual into a programmable practice.