# Activation Probing Framework

Design Document

Version: 20260130

## 1 Goal

Discover natural event spaces in the RNN hidden layer by probing activations at specific positions in the data where we know what's happening semantically.

**Not:** Dump all activations and hope patterns emerge.

**Instead:** Query for specific conditions, examine activations there.

## 2 The Experimental Loop

1. **Hypothesize** an event space (e.g., "previous char was space")
2. **Query** the dataset for positions matching this condition
3. **Probe** the hidden activations at those positions
4. **Analyze** which neurons activate consistently
5. **Compare** to positions where condition is false
6. **Conclude** whether there's a neuron/group encoding this feature

## 3 Query Language

We need to express conditions like:

```
prev_char == ' '           # after space
prev_char in "aeiou"       # after vowel
curr_char == '<'           # start of XML tag
window[-3:] == "the"       # after "the"
in_context("</")           # inside closing tag
```

### 3.1 Primitives

| | |
|---|---|
| `prev_char` | The byte at position $t-1$ |
| `curr_char` | The byte at position $t$ (input to RNN) |
| `next_char` | The byte at position $t+1$ (target) |
| `window[i:j]` | Slice of context around position |
| `position` | Absolute position in file |

### 3.2 Predicates

| | |
|---|---|
| `char == 'x'` | Exact match |
| `char in "set"` | Character class |
| `char.isalpha()` | Letter |
| `char.isdigit()` | Digit |
| `char.isspace()` | Whitespace |
| `window.startswith(s)` | Context match |
| `window.endswith(s)` | Context match |

### 3.3 Compound Queries

```
prev_char == ' ' AND curr_char.isalpha()    # word start
prev_char.isalpha() AND curr_char == ' '    # word end
window[-4:] == "<ref" AND curr_char == '>'  # end of <ref>
```

## 4 Probe Output

For each matching position, record:

```
{
  position: 12345,
  context: "...the <ref>foo</ref> and...",
  curr_char: '<',
  hidden: [0.82, -0.31, 0.95, ...],  # 128 floats
  output_top5: [('r', 0.23), ('/', 0.18), ...]
}
```

## 5 Analysis Tools

### 5.1 Single Query Analysis

Given $N$ matching positions:

- **Mean activation** per neuron: $\bar{h}_j = \frac{1}{N} \sum_i h_j^{(i)}$
- **Std deviation**: How consistent is this neuron?
- **Histogram**: Distribution of activations for each neuron

### 5.2 Contrastive Analysis

Compare query $Q$ (condition true) vs $\bar{Q}$ (condition false):

- **Difference of means**: $\bar{h}_j^Q - \bar{h}_j^{\bar{Q}}$
- **Effect size**: Which neurons differ most between conditions?
- **Selectivity**: Does neuron $j$ activate *only* for $Q$?

A neuron that:

- Has high mean activation for $Q$
- Has low mean activation for $\bar{Q}$
- Has low variance in both cases

...is a good candidate for encoding the feature $Q$.

### 5.3 Clustering

Given multiple queries $Q_1, Q_2, \ldots$:

- Which neurons respond similarly across queries?
- Are there groups of neurons that always co-activate?
- Are there groups that are mutually exclusive?

Mutual exclusivity suggests an event space.

## 6 Expected Lexical Features

Hypotheses to test first (single-char Markov level):

1. `prev_char == ' '` — "after space" neuron(s)
2. `prev_char in "aeiouAEIOU"` — "after vowel"
3. `prev_char in "0123456789"` — "after digit"
4. `curr_char == '<'` — "XML tag start"
5. `prev_char == '\n'` — "start of line"
6. `prev_char.isalpha() and curr_char.isalpha()` — "mid-word"

For each, we expect to find neurons that light up specifically for that condition.

## 7 Expected Event Spaces

If neurons $\{h_3, h_{17}, h_{42}\}$ are:

- Mutually exclusive (never all high together)
- Exhaustive (one is always high)

Then they form an event space.
**Hypothesis:** There's an ES for "character class of previous char":

- $e_{vowel}$: prev was vowel
- $e_{consonant}$: prev was consonant
- $e_{digit}$: prev was digit
- $e_{space}$: prev was space
- $e_{punct}$: prev was punctuation
- $e_{other}$: prev was something else

This would be a 6-way ES encoded across some subset of the 128 neurons.

# 8 CLI Design

```
# Find positions matching a query
./hutter probe --query "prev_char == ' '" --limit 1000 enwik9

# Show activation statistics
./hutter probe --query "prev_char == ' '" --stats enwik9

# Contrastive: compare two conditions
./hutter probe --query "prev_char == ' '" \
               --contrast "prev_char.isalpha()" \
               --stats enwik9

# Dump raw activations for external analysis
./hutter probe --query "prev_char == ' '" --dump activations.csv enwik9

# Interactive mode: try queries, see results
./hutter probe --interactive enwik9
```

# 9 Implementation Phases

## 9.1 Phase 1: Basic Probing

- Load trained model
- Scan file, evaluate simple predicates (exact char matches)
- At matching positions, run forward pass, record hidden state
- Compute mean/std per neuron
- Print top neurons by activation

## 9.2 Phase 2: Contrastive Analysis

- Run two queries (condition true vs false)
- Compute difference of means
- Rank neurons by selectivity
- Visualize with ASCII histogram

## 9.3 Phase 3: ES Discovery

- Run multiple related queries
- Cluster neurons by response pattern
- Identify mutually exclusive groups
- Propose ES partitions

## 9.4 Phase 4: Validation

- Given discovered ES, extract UM patterns
- Compare UM vs RNN accuracy
- Iterate on ES boundaries

## 10 Data Structures

```
// Query result for one position
typedef struct {
    long position;
    unsigned char context[64];  // surrounding bytes
    unsigned char curr_char;
    unsigned char next_char;    // target
    float hidden[HIDDEN_SIZE];  // activation snapshot
} ProbeResult;

// Aggregated statistics
typedef struct {
    int count;
    float mean[HIDDEN_SIZE];
    float std[HIDDEN_SIZE];
    float min[HIDDEN_SIZE];
    float max[HIDDEN_SIZE];
} ProbeStats;
```

## 11 Open Questions

1. How many samples do we need per query for stable statistics?
2. Should we probe pre-tanh or post-tanh activations?
3. How do we handle the recurrent state? (activations depend on history)
4. What's the right similarity metric for clustering neurons?

## 12 Current Status

| | |
|---|---|
| Design document | This file |
| Implementation | Not started |