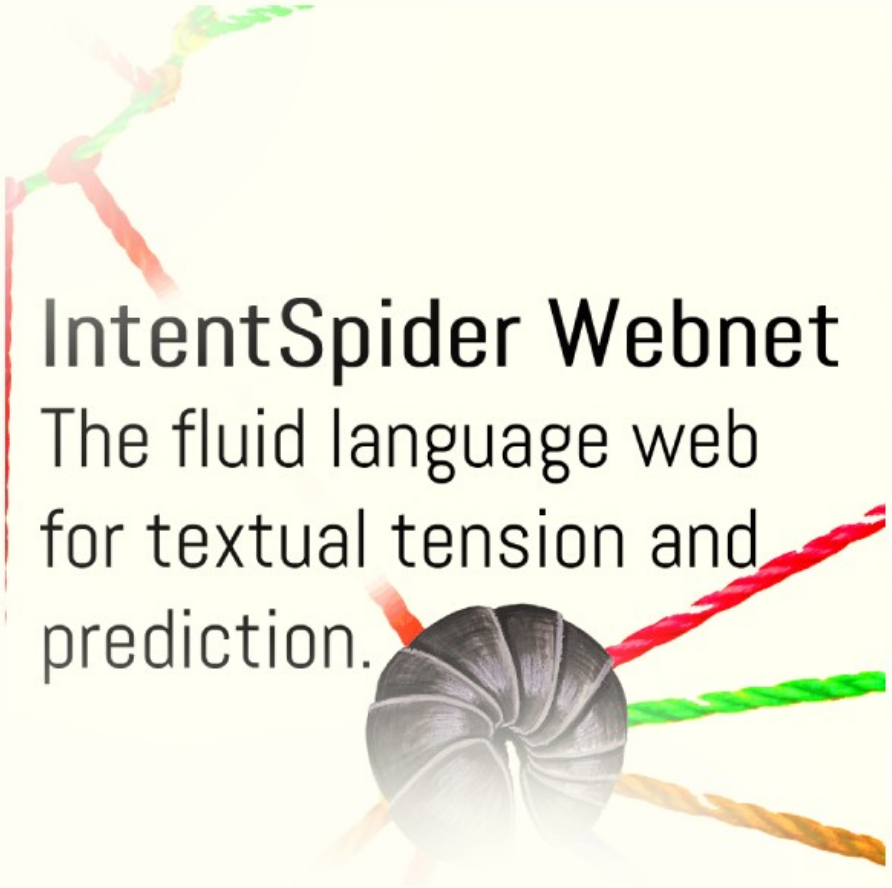
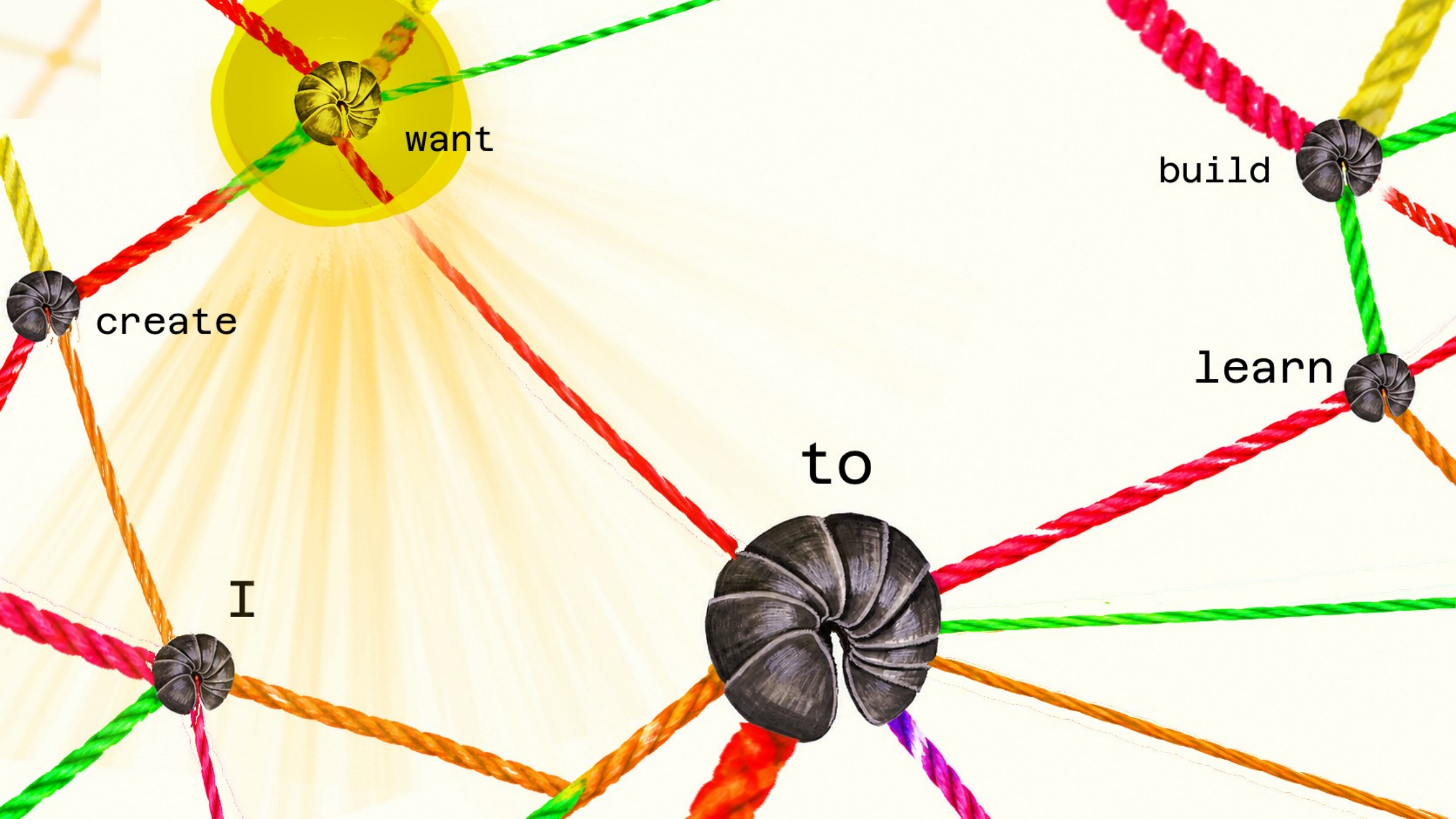
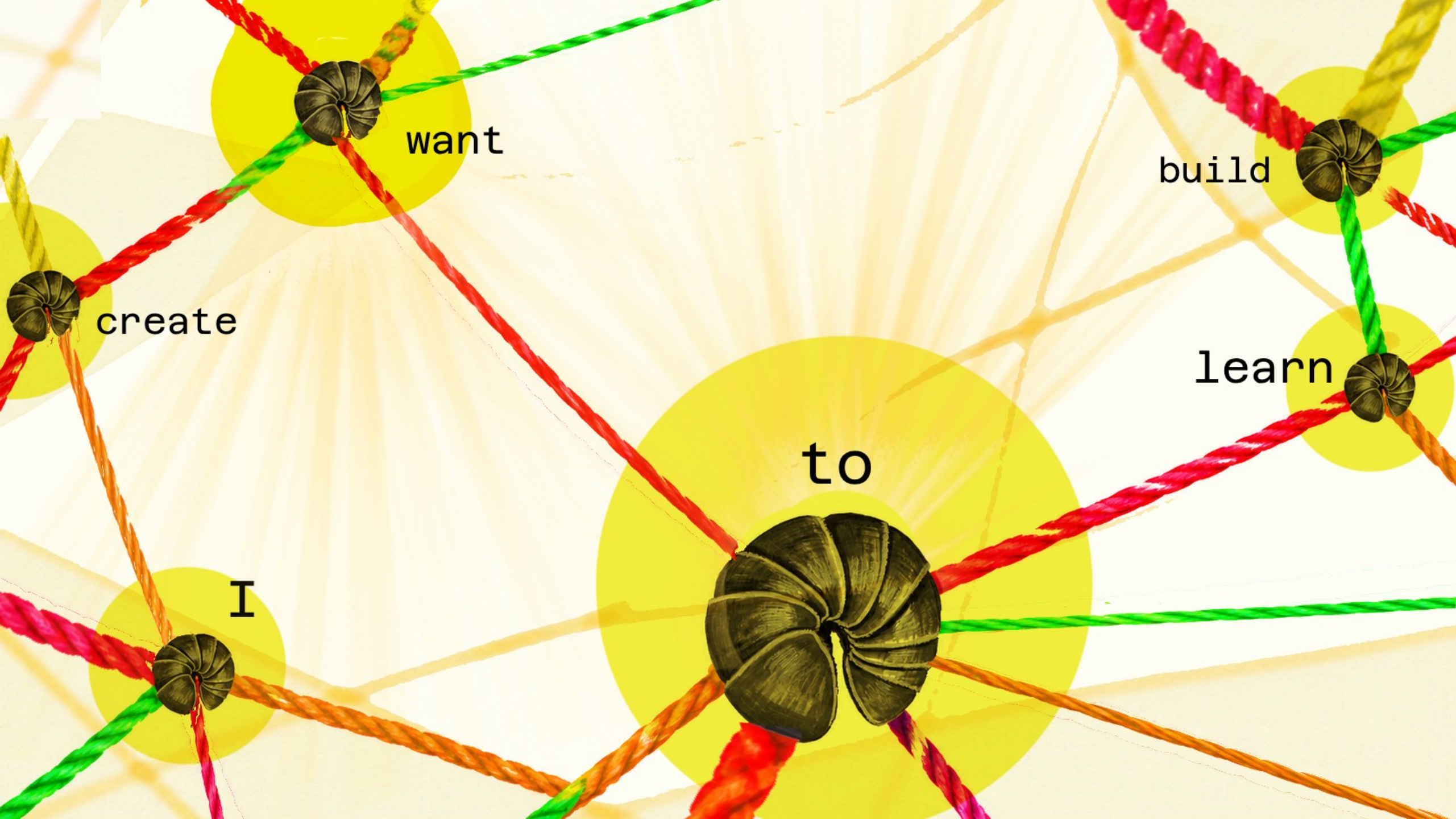


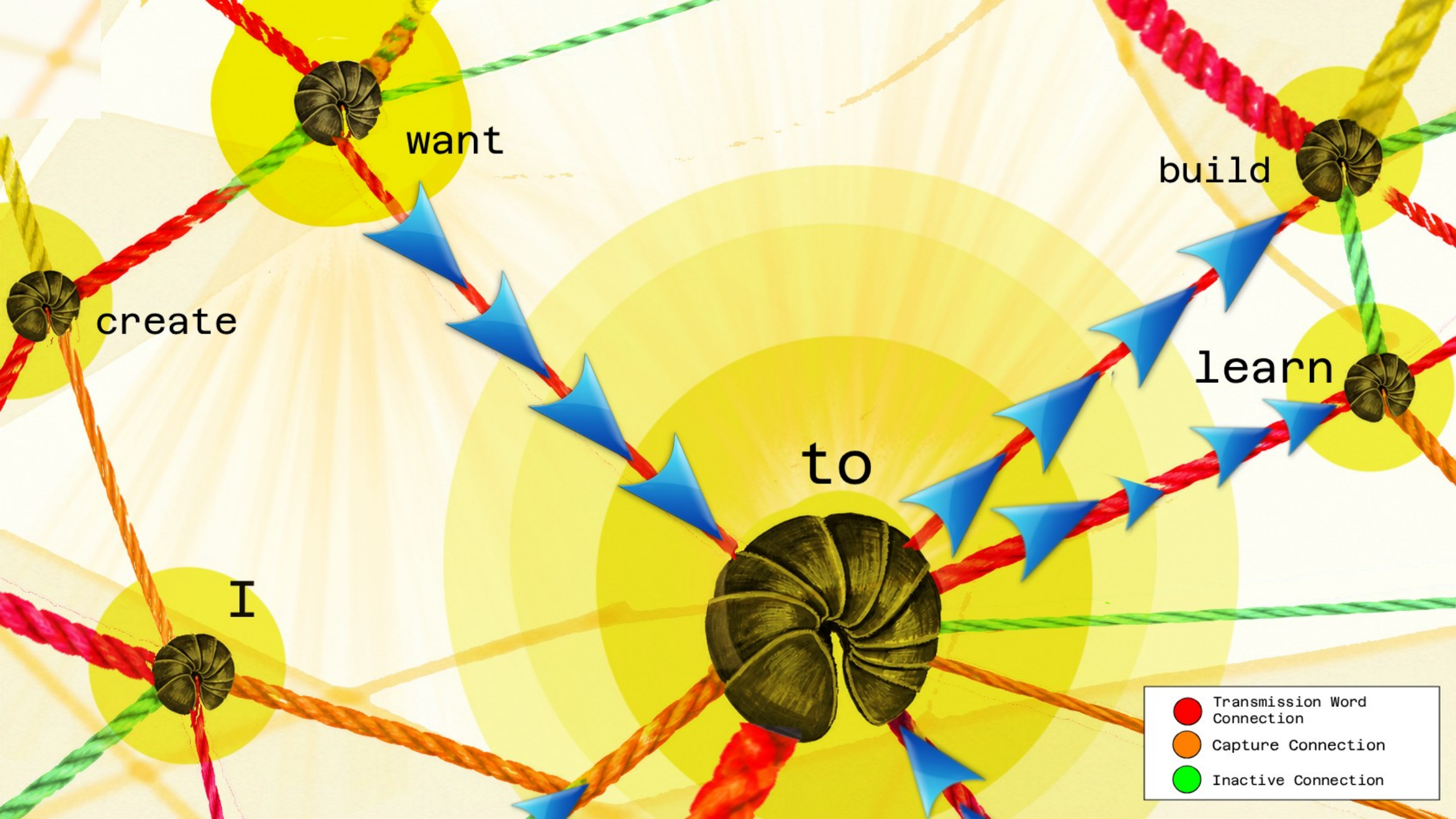

# Presentation

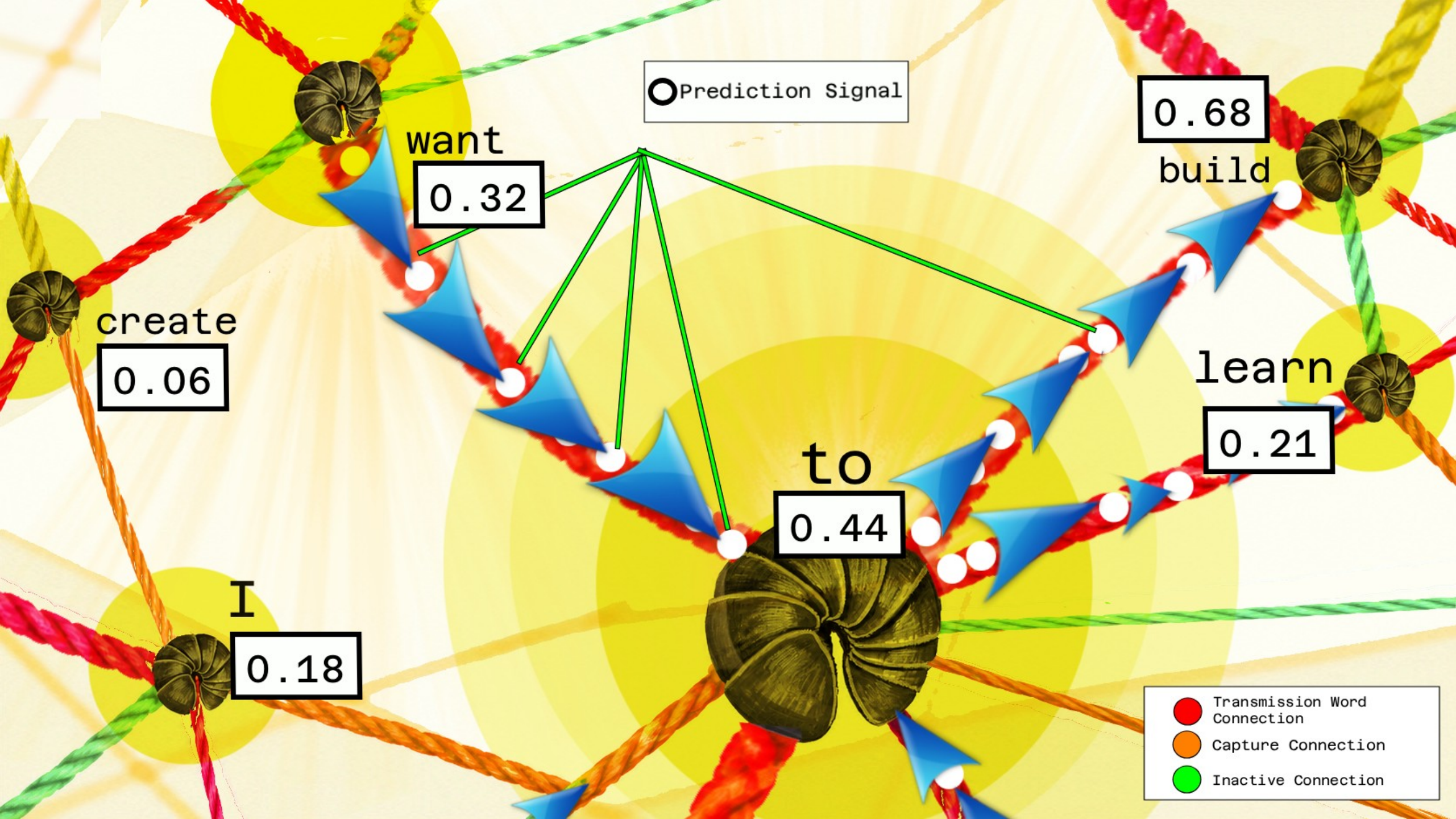
Presenter: Neksha DeSilva

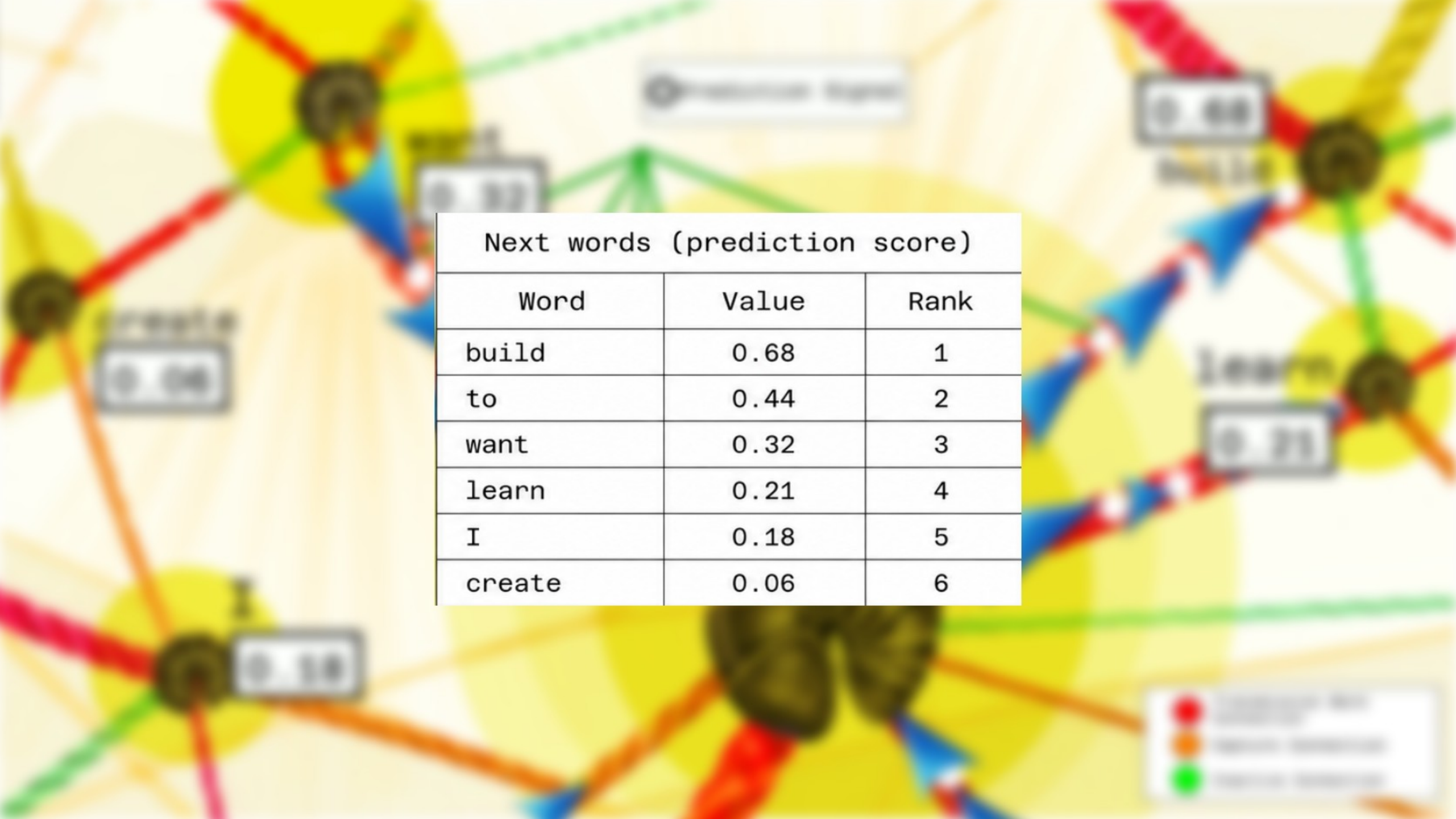



want
build
create
learn
to
I


want
build
create
learn
to
I


want
build
create
learn
to
I
Transmission Word Connection
Capture Connection
Inactive Connection


Prediction Signal
want
0.32
create
0.06
I
0.18
to
0.44
0.68
build
learn
0.21
Transmission Word Connection
Capture Connection
Inactive Connection

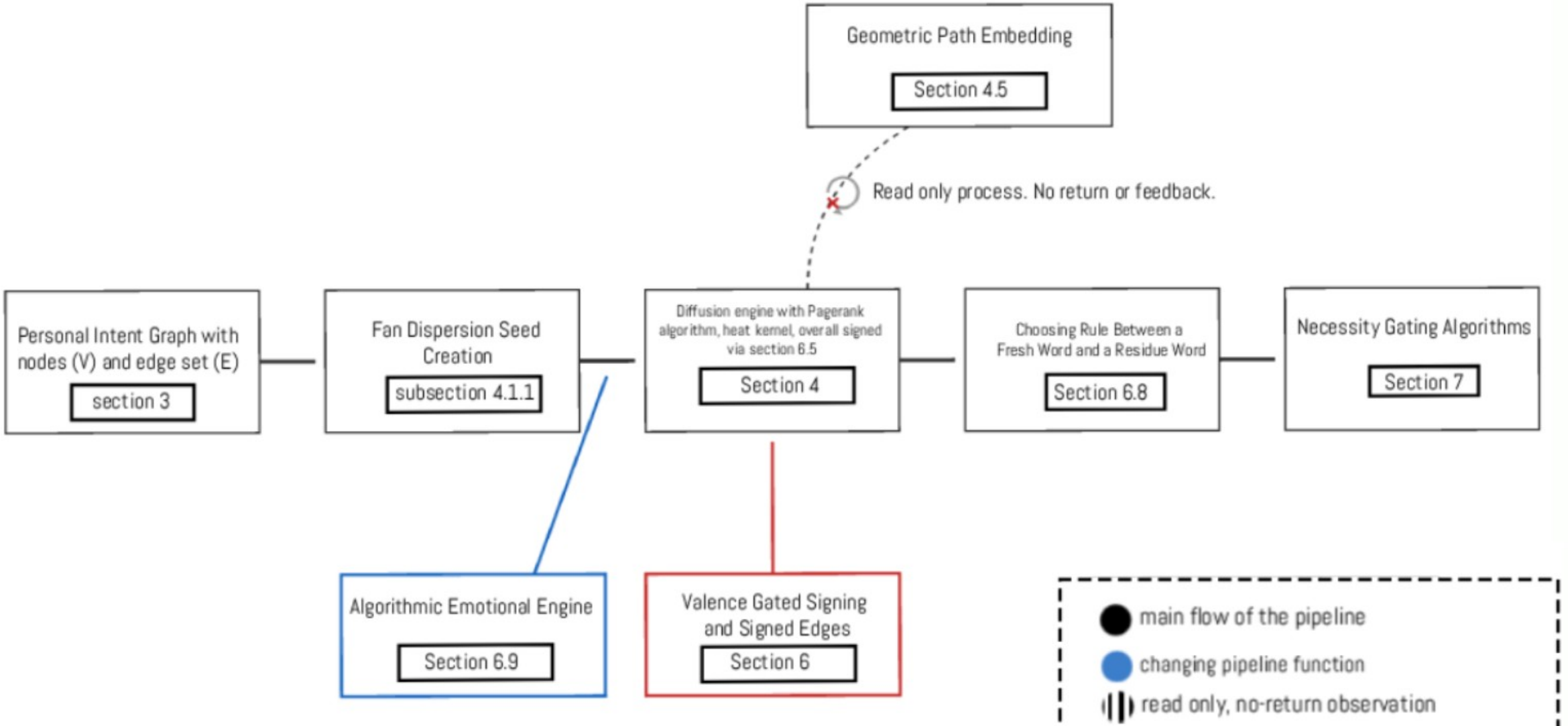


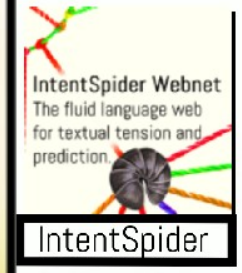
| Next words (prediction score) | | |
|---|---|---|
| Word | Value | Rank |
| build | 0.68 | 1 |
| to | 0.44 | 2 |
| want | 0.32 | 3 |
| learn | 0.21 | 4 |
| I | 0.18 | 5 |
| create | 0.06 | 6 |

IntentSpider Webnet
The fluid language web for textual tension and prediction.
IntentSpider

# Next Word Prediction is an E2E HCI Problem.

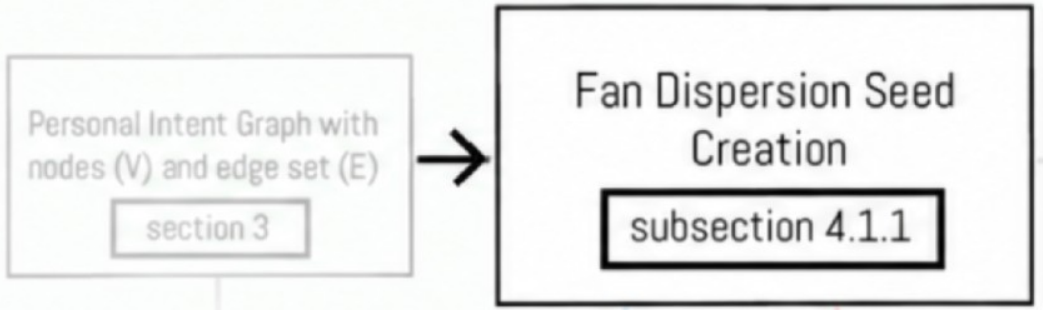
IntentSpider separates context construction, ranking, display, learning and debugging into a personal intent graph.





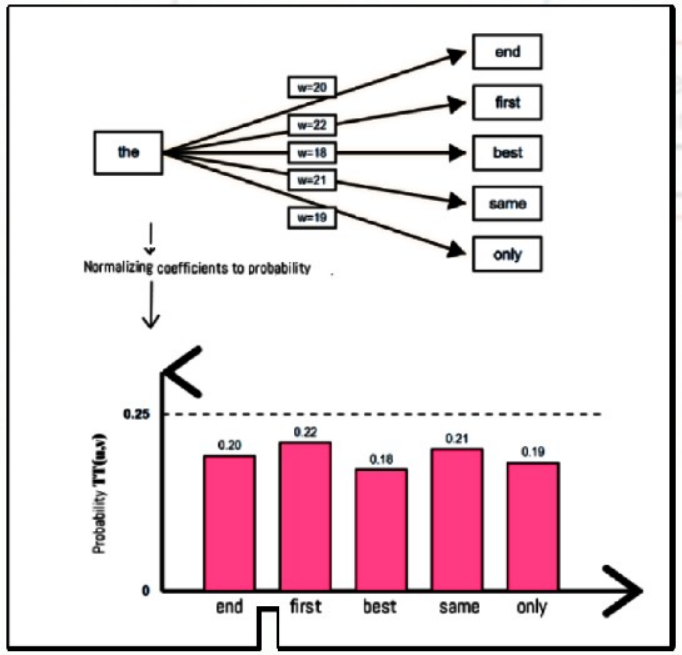

# Next Word Prediction is an E2E HCI Problem.

IntentSpider separates context construction, ranking, display, learning and debugging into a personal intent graph.



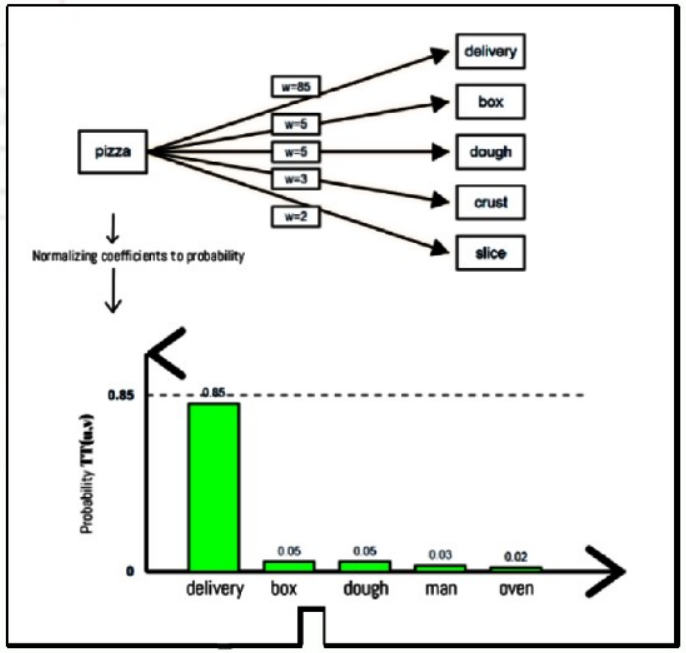

## Context Construction

1. The evaluated process will start from the last 3 committed word set,
2. Recent words are allowed to contribute to the input seed values.



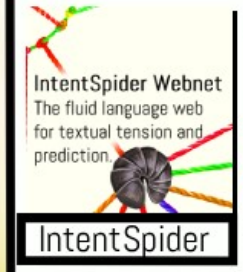

## Broad pattern

Seed input increases




## Cluster pattern

Seed input decreases




# Next Word Prediction is an E2E HCI Problem.

IntentSpider separates context construction, ranking, display, learning and debugging into a personal intent graph.

Personal Intent Graph with nodes (V) and edge set (E)

section 3

The personal intent graph stores the prediction data state of a one human

$$G_{time} = (V_{time}, E_{time})$$

State after event $t$

Fan Dispersion Seed Creation

subsection 4.1.1

Section 4

Choosing Rule Between a Fresh Word and a Residue Word

Section 6.8

Necessity Gating Algori

Section 7

Algorithmic Emotional Engine

Section 6.9

Valence Gated Signing and Signed Edges

Section 6

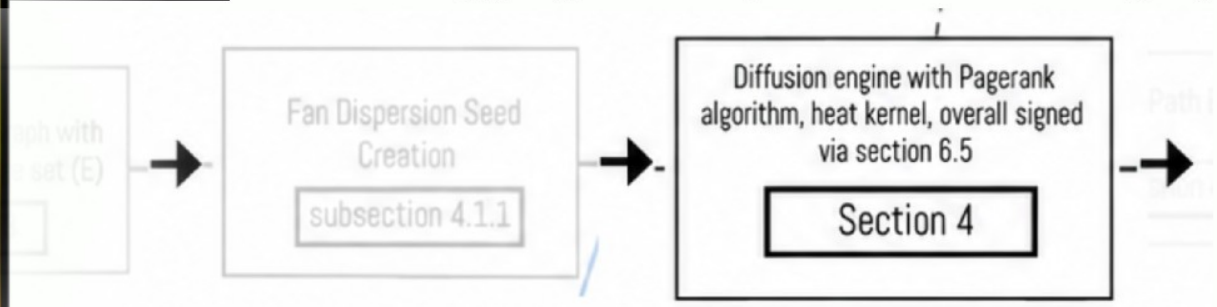
main flow of the pipeline

changing pipeline function

read only, no-return observation

# Next Word Prediction is an E2E HCI Problem.

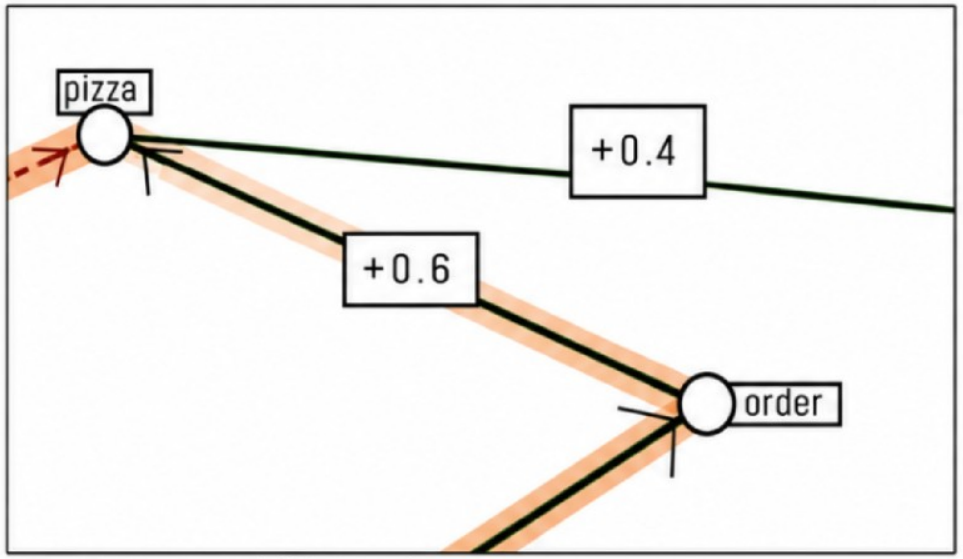
IntentSpider separates context construction, ranking, display, learning and debugging into a personal intent graph.




## Graph Ranking Within Graphs

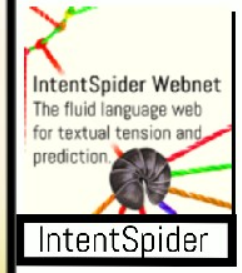
1. Local push process will be executed after the seed input is created.
2. Before ranking process, context words that arrived recently will be removed
3. Other (+) positive words will be ranked by the valiue assigned in the diffusion process.

Path of this diffusion process (geo. path)



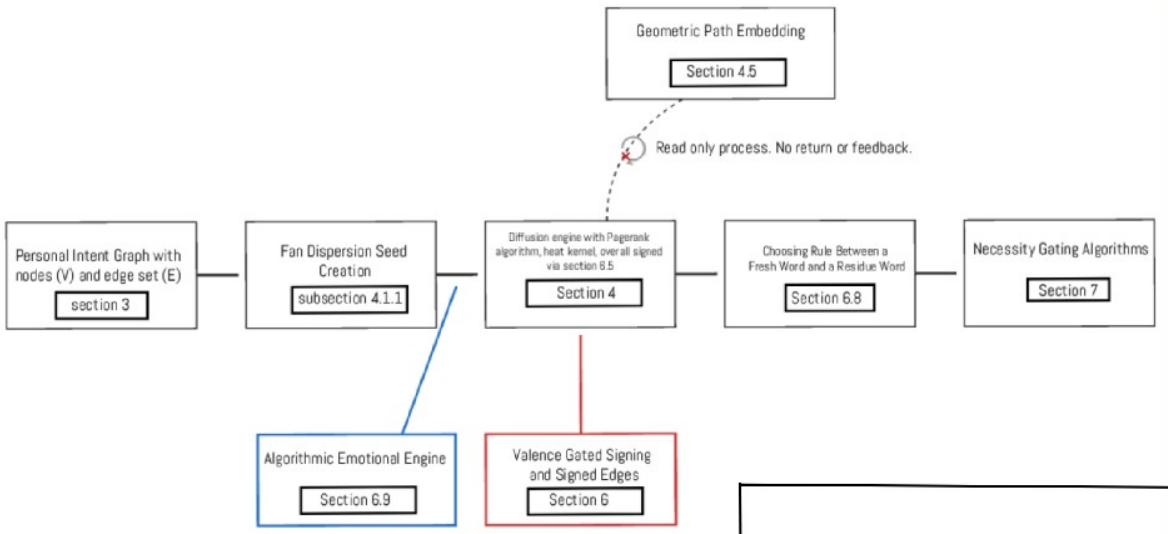

Notes:

1. TF type (transmission) connections will forward diffusion values
2. Then, postitive (+) words will be ranked by diffusion values




# Next Word Prediction is an E2E HCI Problem.

IntentSpider separates context construction, ranking, display, learning and debugging into a personal intent graph.



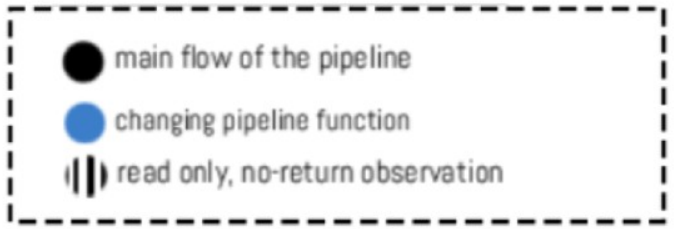

## Graph learning process

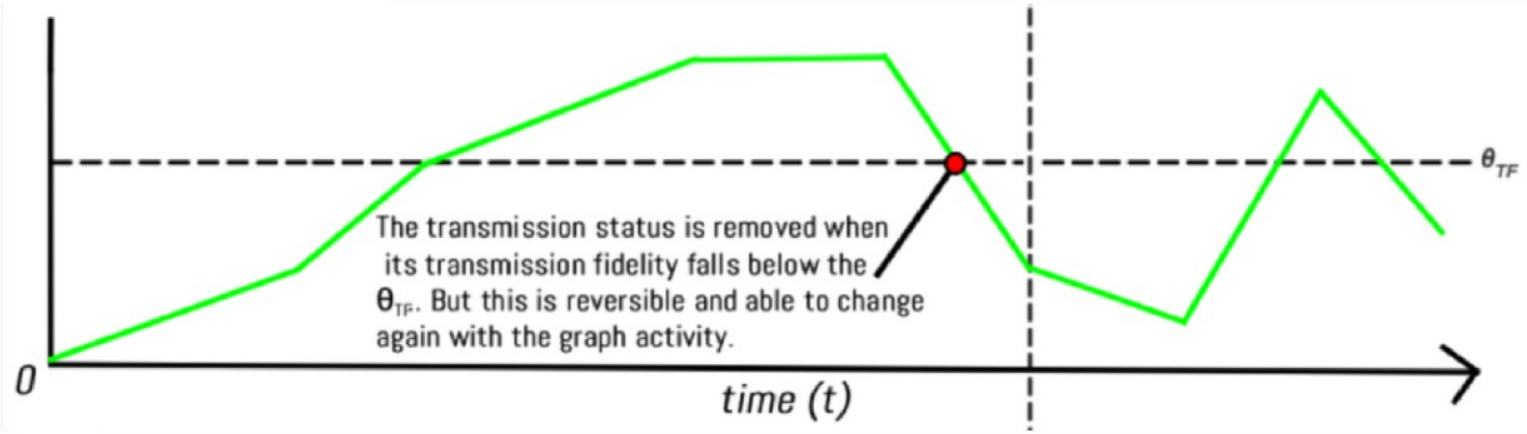
| Rank Results | |
|---|---|
| Word the user selected | 1 |
| Other ranked words on the graph | 0 |

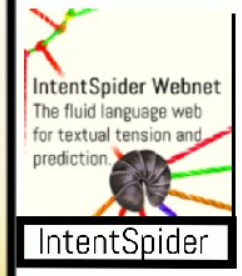
→ $TF_{value}$ *update ()*

| Connection state is allowed to change |
|---|
| TF type ⇌ Capture type |

Note: This is a reversible state

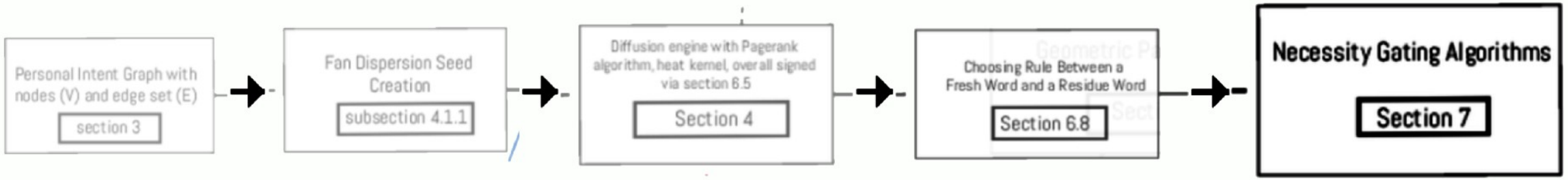





# Connections, and Their Different Roles in Prediction

IntentSpider separates context construction, ranking, display, learning and debugging into a personal intent graph.




🔴 BLOCK;

*Pseudocode*

🟢 RELEASE;

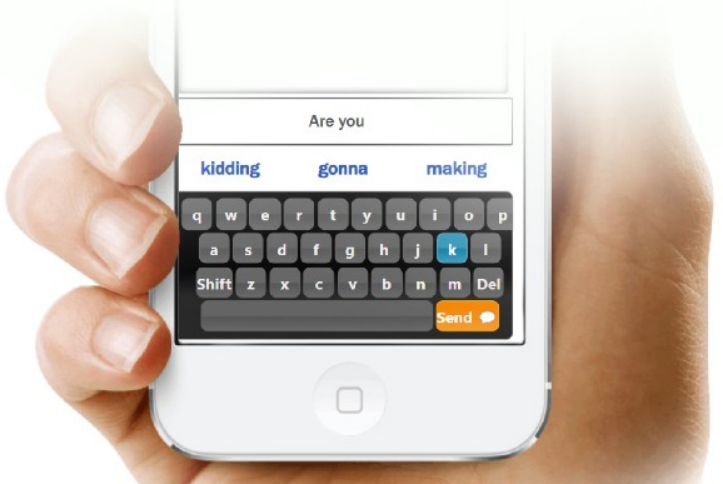
suggestion row remains "hidden"

## Internal hidden state




suggestion row is "visible"

## Display state



# Connections, and Their Different Roles in Prediction

Every conneciton will store a connection value and a TF value. The connection value depicts the connection relationship. TF value controls the diffusion forwarding logic.

## Transmission Type (TF Type)

This type forwards diffusion coefficient X

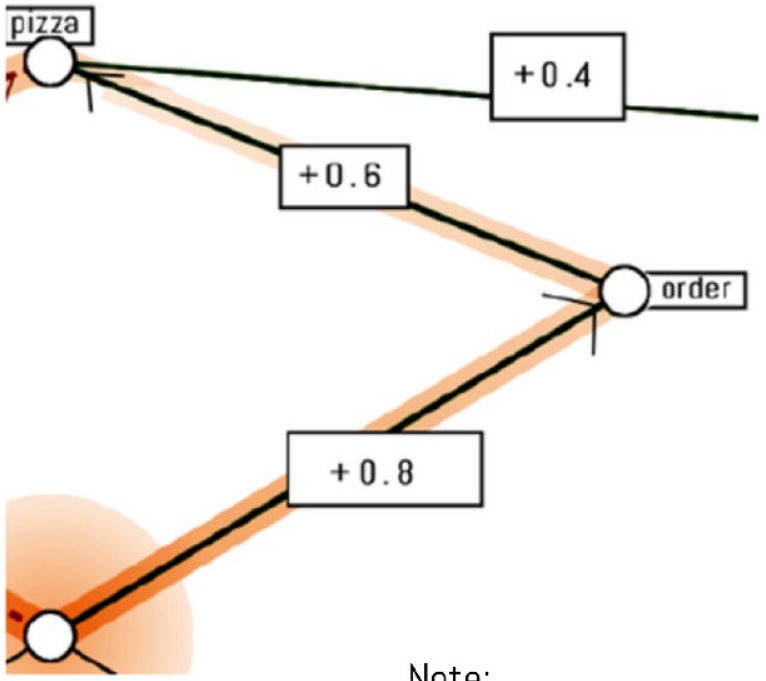



Fig 2.

Note:
1. Storing the ordinary connection value
2. Functions in the idle state

## Capture

This type does not forward diffusion coefficient X

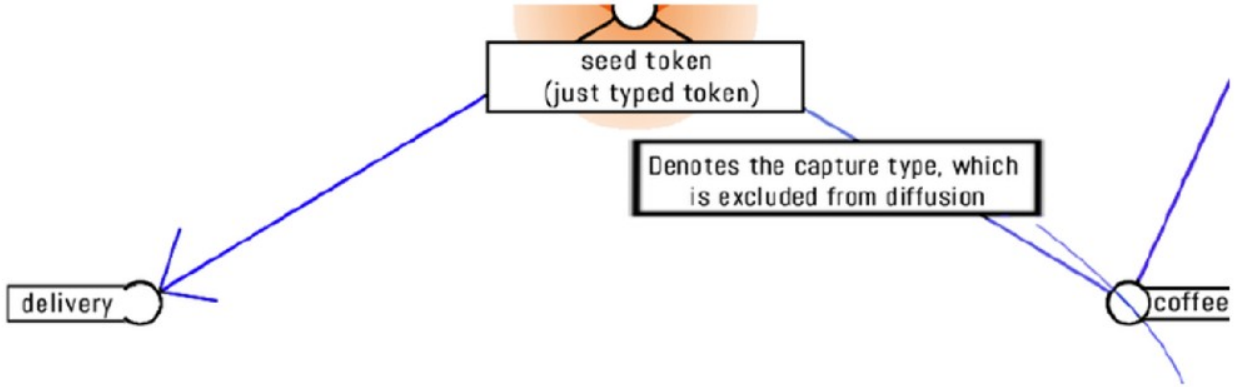



Fig 2.

Note:
1. Conditional logic for forwarding diffusion coefficients
2. Functions in the idle state

IntentSpider Webnet
The fluid language web for textual tension and prediction.
IntentSpider

# Word Ranking Process

After executing the local push algorithm, input seed value will be created.

## 1 Step 01

### Executing Local Push - TF Method

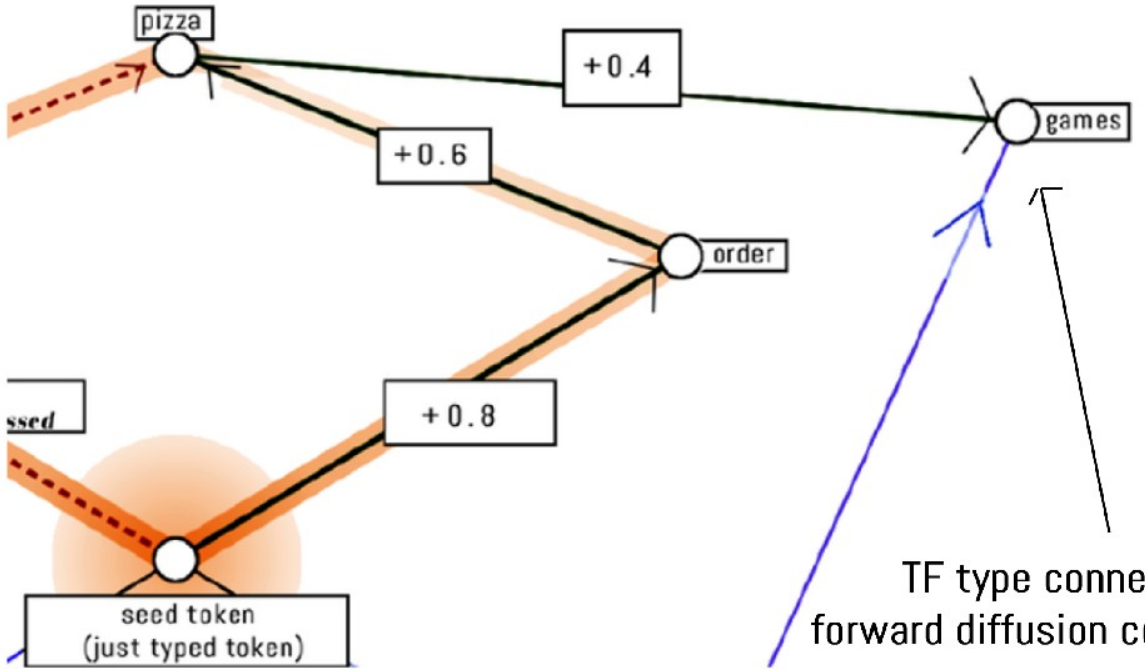



Fig 2

Seed Input → Push Process → Ranking

## 2 Step 02

### 1 Ranking Process

Recent context words will be removed before the ranking process

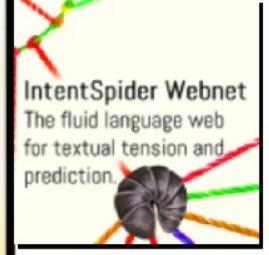
### 2 Other Positive Words

These positive words will be ranked by diffusion value

XXXX XXXXX XXXXX

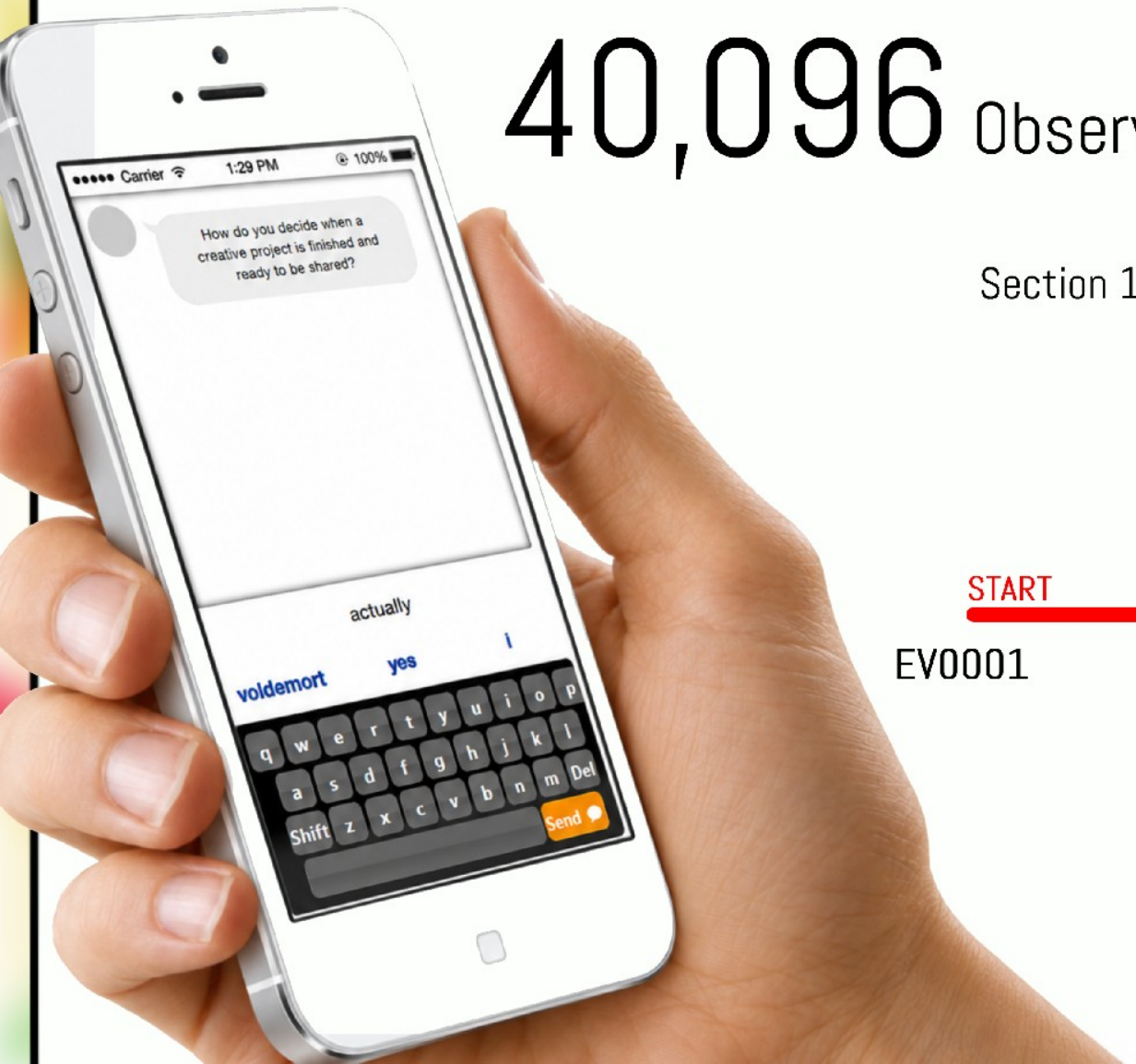
Internal ranked row
(not "RELEASED;" yet)

THEN: Conditional "DISPLAY;" Decision

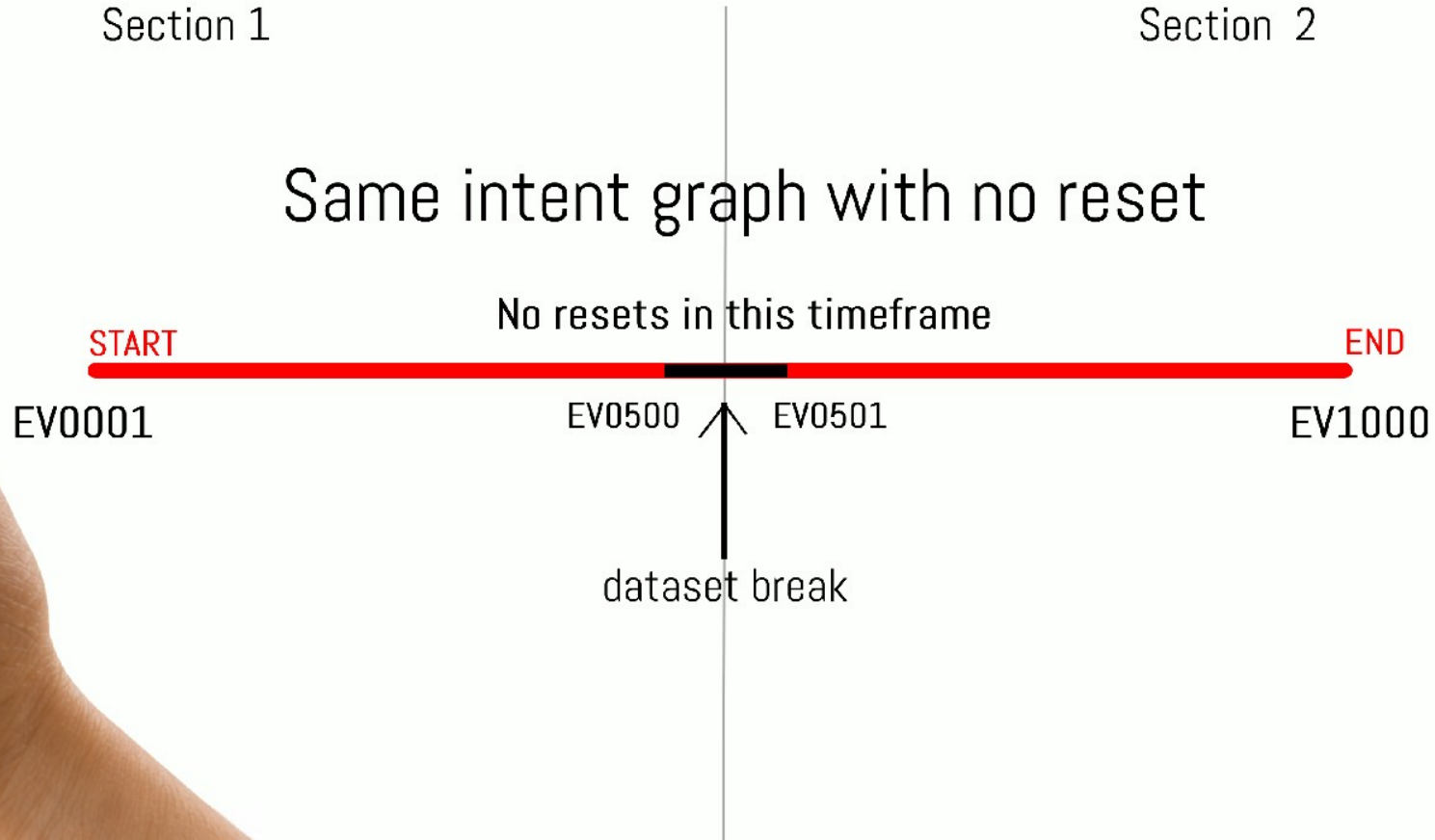


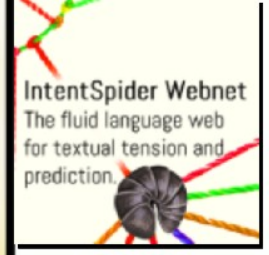

# Experiment: 1,000 Continuous Prediction Events

Evaluation test 0001 to test 1000 continuous executions without resetting the graph




## 40,096 Observations at the starting state






# Experiment: 1,000 Continuous Prediction Events

Evaluation test 0001 to test 1000 continuous executions without resetting the graph



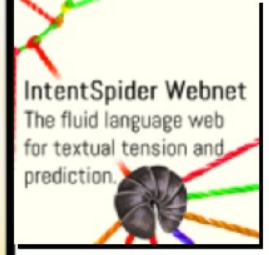

**40,096** Observations at the starting state

**1,000** Prediction event count (Overall)

**5** Different prediction architechtures

**65** Repetitions

**105** Waste events




# Experiment: Same State, Same Target Order, One Process Changed.

Every execution output is a result of the dynamic graph state at the moment of execution

## Same Saved Graph State

Previously recorded intent graph state
loaded into the IntenSpider engine

+

## Same Target Order for Words

Entering the same 500 eval events again

## One Process Changed.

Same starting graph state, same target order shared with others
and isolated process change

Stats:

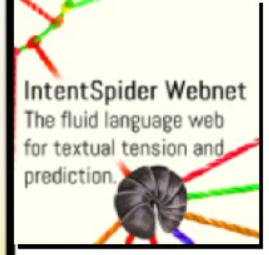
3,198 Prediction rows from GBoard, Swiftkey, OpenBoard and Samsung Keyboard.

33,000 Eval pair states (from count selector and word count prediction func)

Ran online on,

Google colab

NVIDIA TESLA




# The Evolving Intent Graph, Predicts!

Execution label no. EV0001 to EV1000

notes

In here, 1 prediction event correlates to a valid opportunity for predicting the next word
This execution is done from EV0001 to EV1000.

---

# 13.2%

## First rank accuracy

In cases where the target word is positioned in the first position (Rank 1)

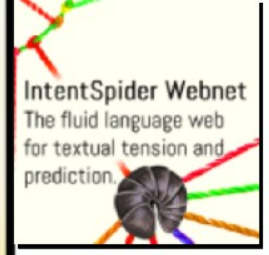
132 correct events out of 1,000 total

# 25.4%

## Top 3 accuracy

In cases where the target word is positioned within 1st to 3rd position. (1 to 3 rank)

254 correct events out of 1,000 total




# The Evolving Intent Graph, Predicts!

Execution label no. EV0001 to EV1000

## Main result

13.2% 132/1000 First rank accuracy

25.4% 254/1000 Top 3 accuracy

Highest Top 3 Accuracy Score:

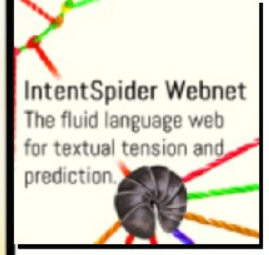
15% Kneser Ney

vs

25.4% IntentSpider Webnet (Top)

notes

1. Tested comparison methods included most recent word count and Kneser Ney
2. Highest reported top 3 result was 15%.
3. Please note. These methods used different starting information from IntentSpider and were not same state process comparisons as others.



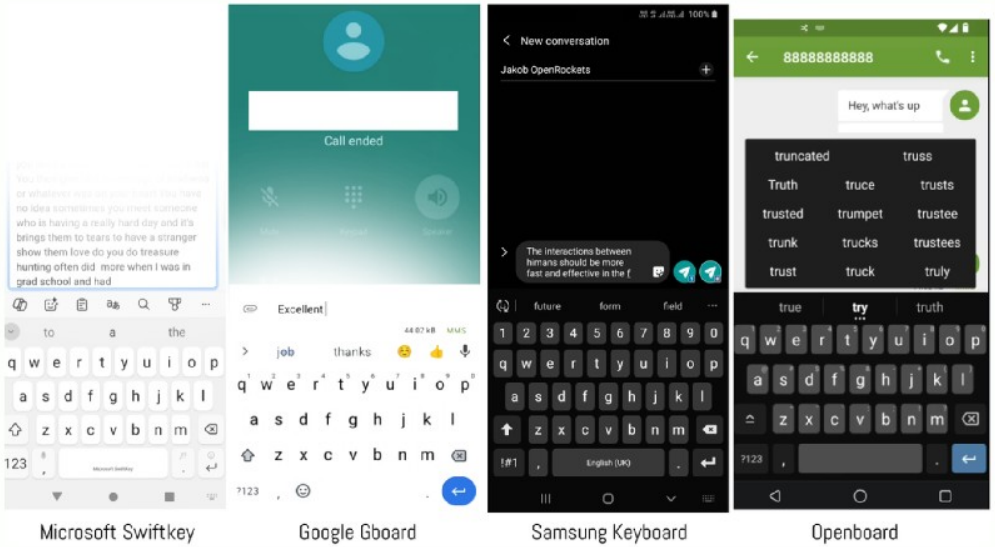

# The Evolving Intent Graph, Predicts!

Execution label no. EV0001 to EV1000

## Main result

**13.2%** 132/1000 First rank accuracy

**25.4%** 254/1000 Top 3 accuracy




Microsoft Swiftkey | Google Gboard | Samsung Keyboard | Openboard

## Other Keyboard Comparison

**4 OUT OF 4**

IntentSpider measured higher in

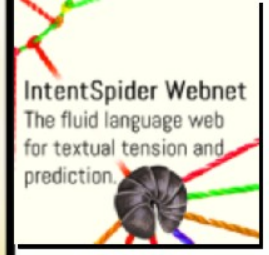
across all 4 comparisons on external prediction systems

Both 1st rank and Top 3




Table no. 3

| Product | $N$ | First rank | $\Delta_1$ | Top 3 | $\Delta_3$ |
|---|---|---|---|---|---|
| Gboard | 394 | 17.77/7.61 | +10.15 | 33.25/14.47 | +18.78 |
| Samsung Keyboard | 934 | 14.03/0.64 | +13.38 | 27.52/2.25 | +25.27 |
| Microsoft SwiftKey | 955 | 14.14/2.20 | +11.94 | 27.33/6.18 | +21.15 |
| OpenBoard | 915 | 14.75/0.44 | +14.32 | 28.09/5.68 | +22.4 |




# The Evolving Intent Graph, Predicts!

Execution label no. EV0001 to EV1000

## Main result

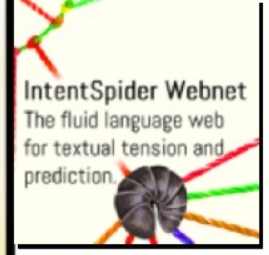
13.2% 🔑 132/1000 First rank accuracy

25.4% 🔑 254/1000 Top 3 accuracy

Table no. 3

| Product | $N$ | First rank | $\Delta_1$ | Top 3 | $\Delta_3$ |
|---|---|---|---|---|---|
| Gboard | 394 | 17.77/7.61 | +10.15 | 33.25/14.47 | +18.78 |
| Samsung Keyboard | 934 | 14.03/0.64 | +13.38 | 27.52/2.25 | +25.27 |
| Microsoft SwiftKey | 955 | 14.14/2.20 | +11.94 | 27.33/6.18 | +21.15 |
| OpenBoard | 915 | 14.75/0.44 | +14.32 | 28.09/5.68 | +22.4 |

notes




# What changed the ranking most?

Same state, target order and one process change.

Focus

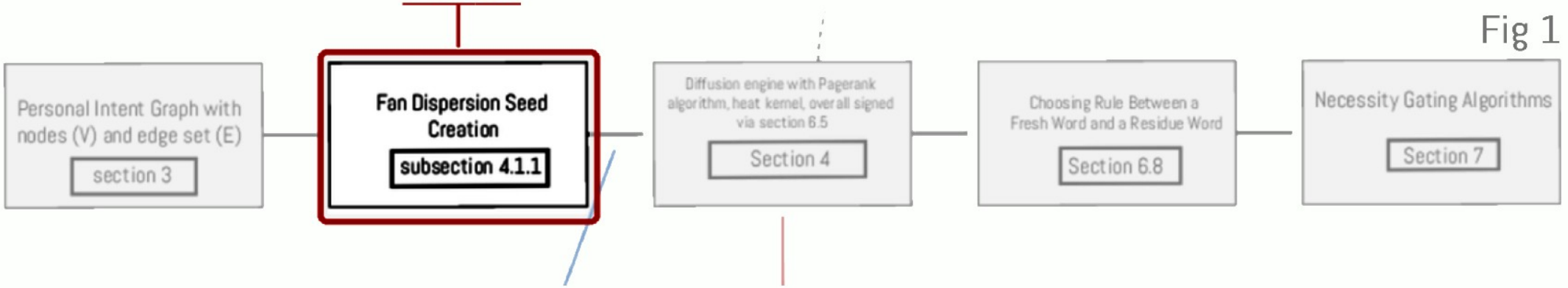



Fig 1

Reference
k value
3

Last 3 words
(committed words only )

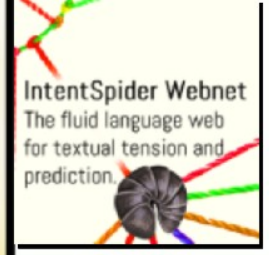
The depth of context changed ++

Reference
k value
1

Most recent word only

notes

1. Last 3 committed words and most recent word both are taken from the seed input values in fan dispersion algorithm. TF will be active and dffusion and the signed value process will be active




# What changed the ranking most?

Same state, target order and one process change.

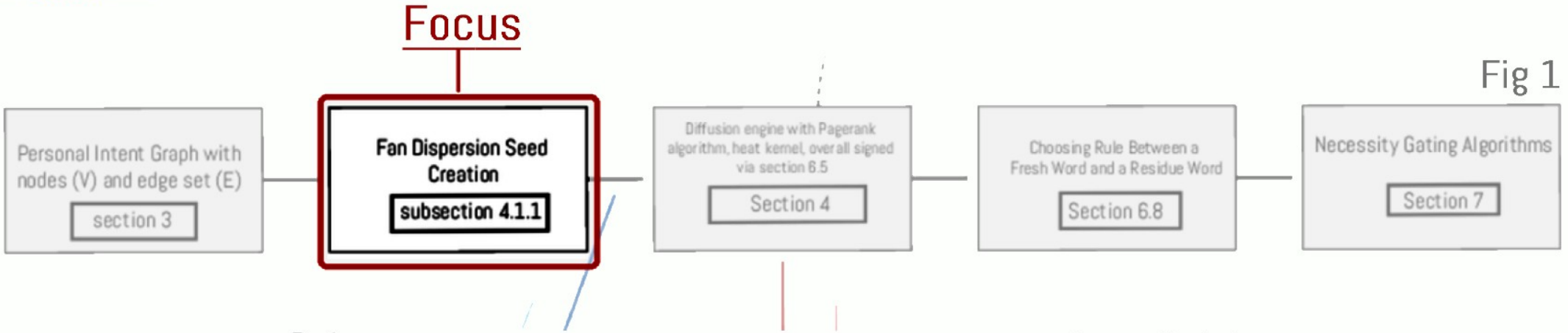



Fig 1

Reference
k value
3
Last 3 committed words

The depth of context changed ++

Controlled change
k value
1
Most recent word only

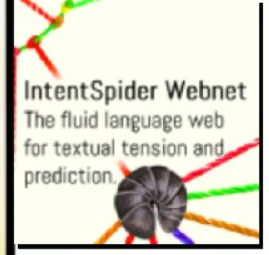
24.6%
Top 3 ranking (R3)

42.6% TO 49.2%
Top 3 ranking range (R3)

The ranking itself changed.

notes

1. Both conditions use the same recorded state and the same target order. Only the seed input depth changes.
2. TF, diffusion and the signed value process will be active.




# Changing Display Output Without Changing Prediction States

Same state, target order and one process change.

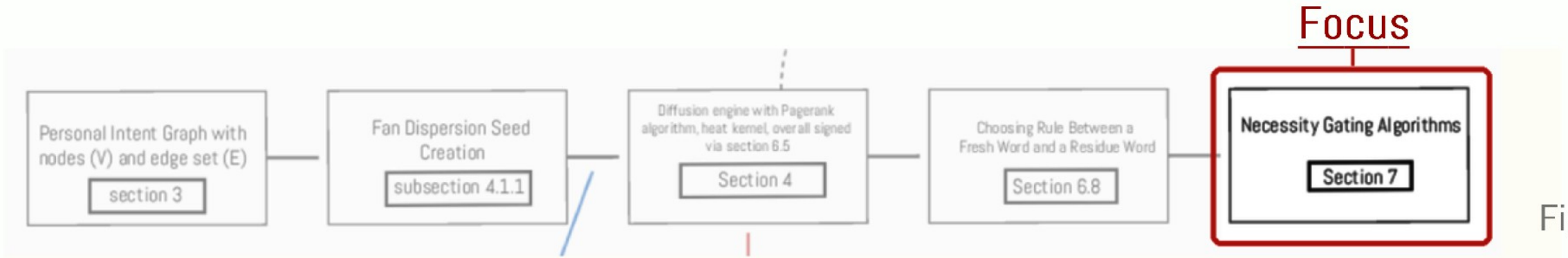



Fig 1

Reference

## Normal Gate

Current gating decision

$$g_{time} = G(r_{time}, S_{time})$$

display conditional logic changes

Controlled change

## Open Gate

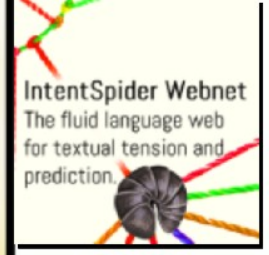
Display every row that is !null.

$$r_{time} \neq \varnothing$$

---

notes

1. The recorded graph current state in 500 event target order and ranking process will not be changed.
2. Only one process changes while the graph update input will not be changed.




# All Rankings are Identical, but Visibility Has Changed.

This Gate Open and Replaying Process will only change the displayability. It does not affect the 500-event ranking order or sequence.

## 500 / 500 The entire ranked row array is identical.

### Reference Gate

**94** Ranked, But Hidden Rows




### Open Gate

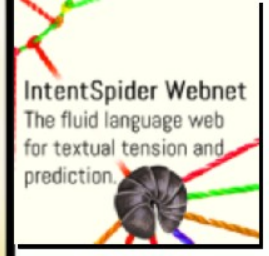
**59** Top 3 Correct, **Force** Displayed

Open Gate




notes

1. Correct predictions were already predicted before the display decision at the time the rank state was unchanged
2. Two seperate systems showed the same display seperating states in 33,000 paired event states




# From Keypress to Suggestion under 16.67 milliseconds!

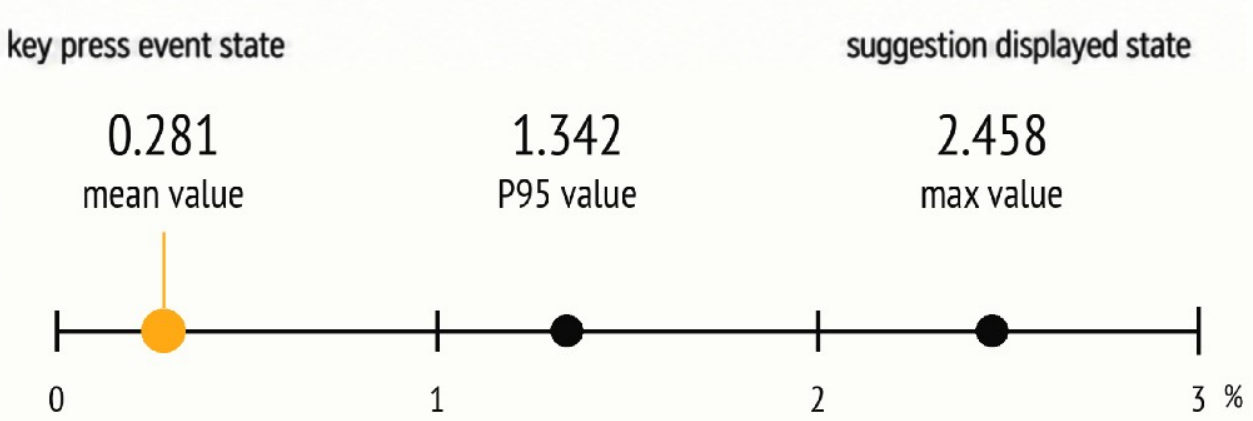
This is Measured end-to-end from keyboard input to C++ prediction to a final, displayed suggestion.




IntentSpider Research Playground

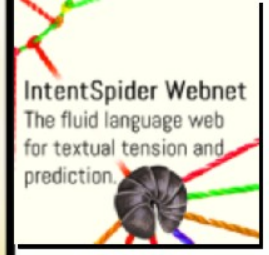
https://intentspider.github.io/website/playground

## The Complete Interaction Path




- 50 out of 50 events finished below 16.67 miliseconds
- Browser: Microsoft Edge
- Operating System; Windows 11 Pro
- Processor: Intel i71165G7
- Test Runtime Duration: 1 min and 30 seconds (90s)

notes




# Within 30 Minutes, IntentSpider Only Consumed Approx. 0.281% CPU (mean).

30 minute human typing experiment data measured using
Microsoft Windows Statistics, on IntentSpider Webnet.

## Runtime Test Duration (Measured)

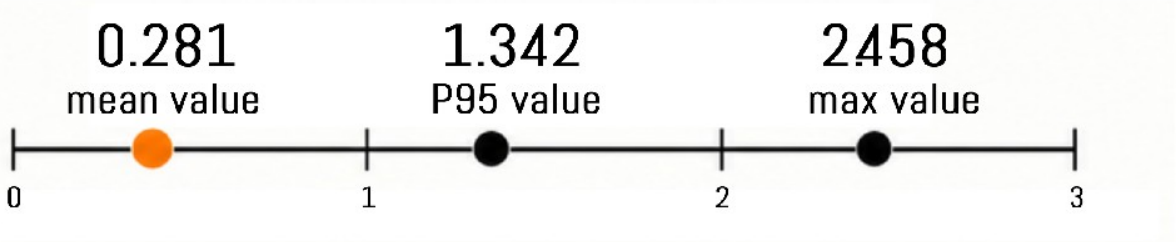
# 30 minutes




IntentSpider Research Playground
https://intentspider.github.io/website/playground

## CPU Usage (Measured)

# 0.281% (complete test runtime)

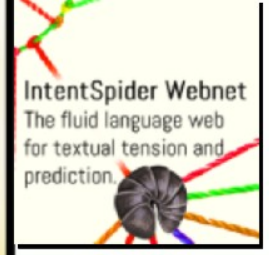



Personal Intent Graph Size: 2.316 Megabytes

IntentSpider KV Loaded State
(Globalperson Mode Only) : 2.255 Megabytes

notes

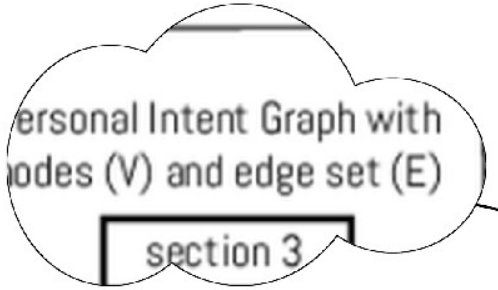
The CPU workload is subjectively very low in this experiment, which supports its longterm adoption and practical use.



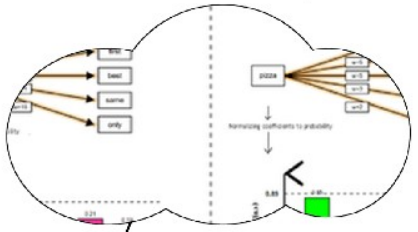

# In A One Personal Intent Graph, Six Processes Can be Tested.

These 6 questions shape the next word prediction and the user experience.

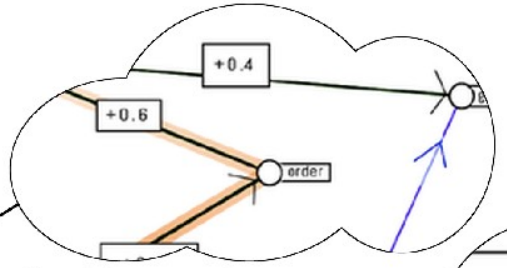
# Evolving, Personal, Intent Graph



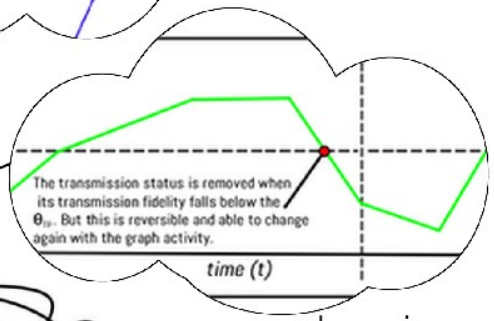

Personal State



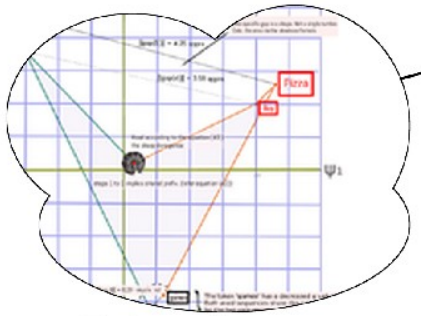
Getting Context




Ranking words



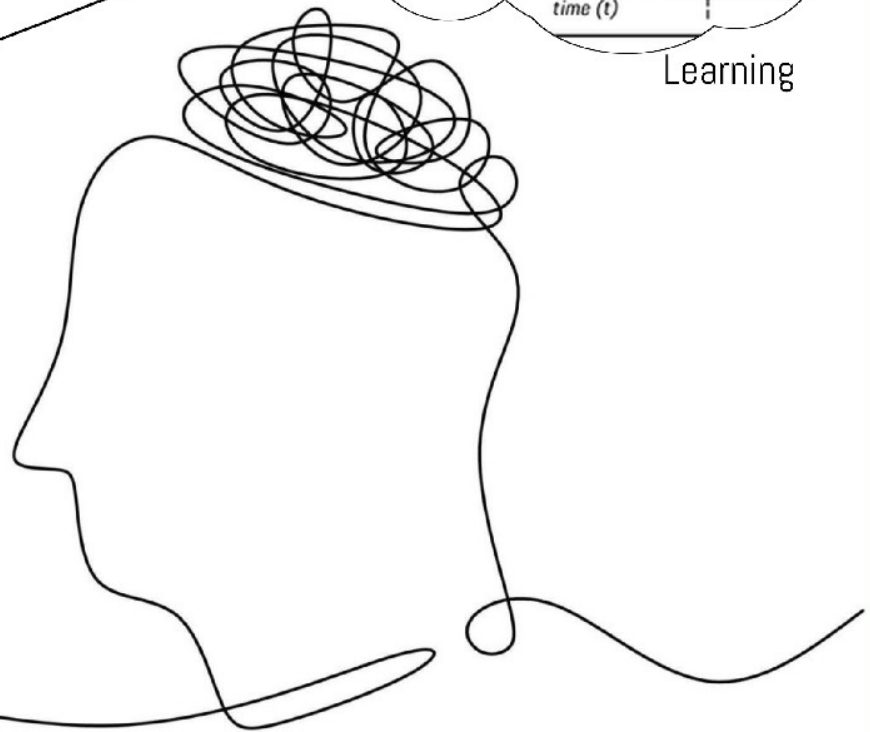

Learning



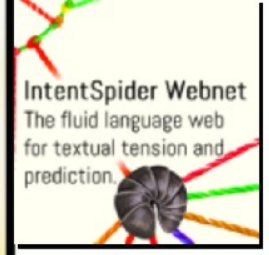
Debugging




Displaying

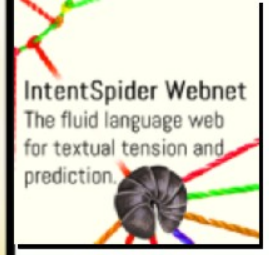


notes




We believe a predictive result
is valuable, only when we can analyze
the underlying changes rather
than the final outcomes.

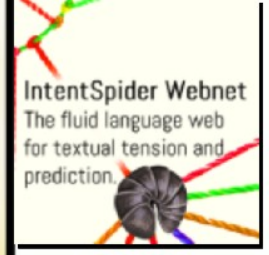
notes




We are actively exploring how IntentSpider's intuitive architecture and high efficiency can extend to other languages.

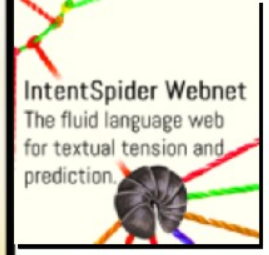
The next step is expansion beyond English into latin and roman rooted languages such as German, Spanish, and more,

notes



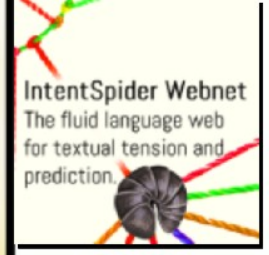

With the goal of making personal intelligence
more privacy focused, safe,
and computationally efficient.

notes



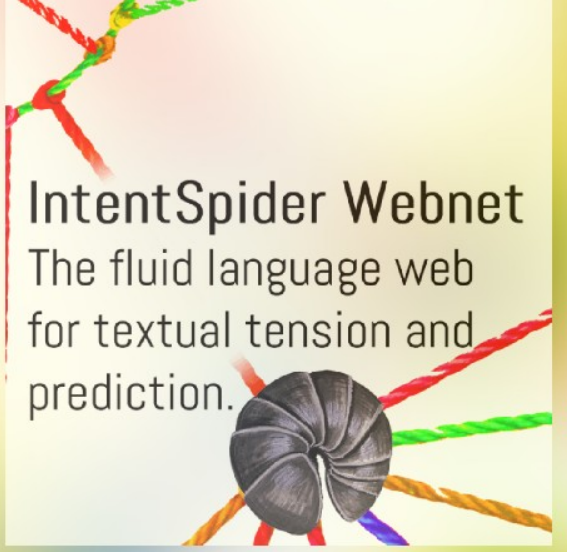

IntentSpider is only beginning.
Next word prediction is the
first step toward discovering
the broadest patterns in language, nature
and beyond.

notes




# IntentSpider is the Fluid Language Web for Textual Tension and Prediction.

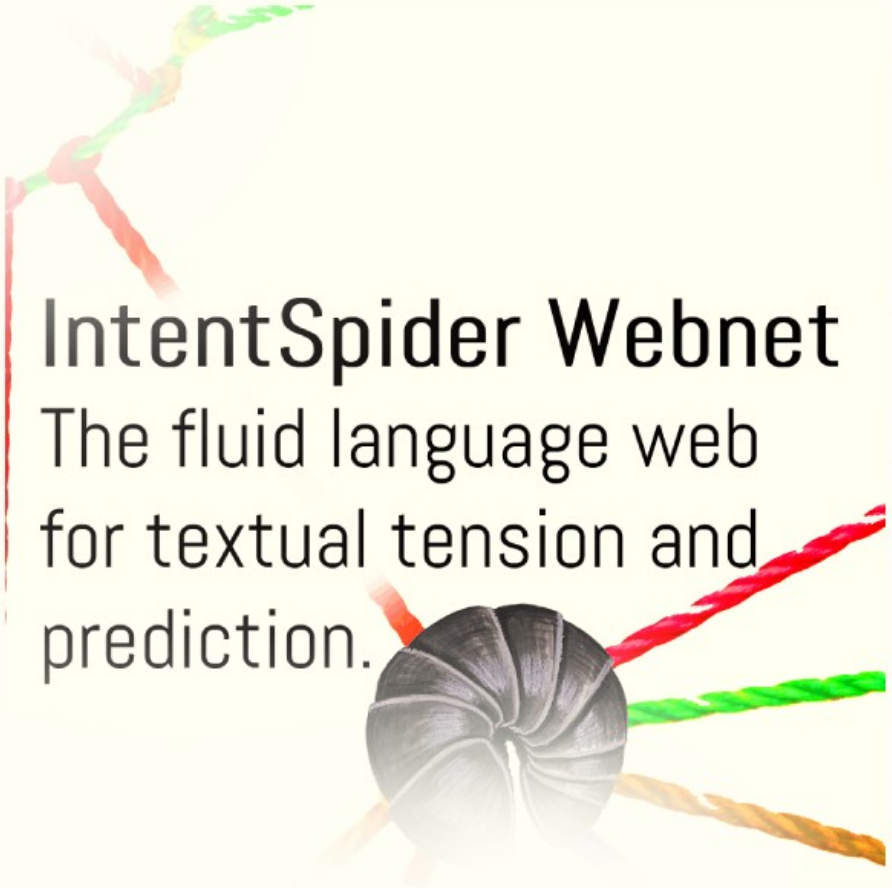
notes




Thank you.

Playground: IntentSpider.GitHub.io
Essay: IntentSpider.GitHub.io/website/essay
Paper: IntentSpider.GitHub.io/website/paper

Presented by: Neksha DeSilva




# Presentation

Presenter: Neksha DeSilva