Protocol Documentation:

Overview.

Knowledge-sharing system. It is an effective tool for any web community that needs to capture and share information, and is unique in that the knowledgebase grows smarter every time it's used. It only knows what you teach it, but it's very easy to teach.

The *self-organizing neural mapper,* allows to incorporate terms used in each search into a contextual map of the answer itself, continually improving ability to derive contextual information from a given search.

Neural Network approach allow to create flexible Data structure, that can be used in FAQ systems , Search engines etc.

User ask system with natural language

System will search the answer and return results ordered by relevance points .

User can create extraction rules, so the answers will be more relevant.

Teaching :

User create record with natural language

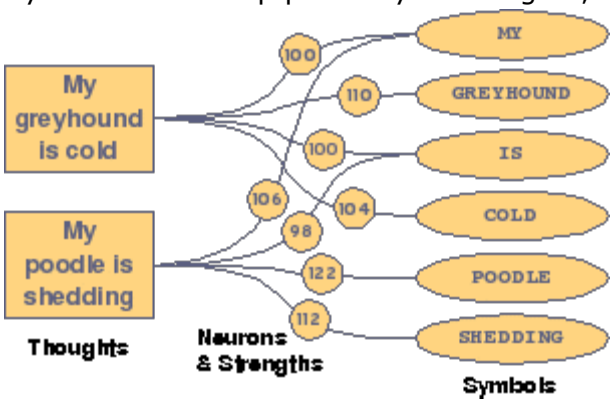
User can assign new meaning to the word (create synonyms)

User can Give feedback to the system so the last answer will be more relevant for the next use.

User can create extraction rules.

Thoughts, Symbols, and Neurons

System is made up primarily of thoughts, **symbols,** and **neurons.**



A *thought* is a specific answer to a specific question, and includes a *summary* (the question being answered) and a *detail*(the answer itself, in text or HTML format). To simplify the example, we'll only use the summary and ignore the detail.

A *symbol* is an alphanumeric term: a word, a number or some combination of the two. System stores one copy of each unique symbol that ever occurs in a thought or is asked during a search. *Examples of symbols*: DOG, ERROR1274, 256

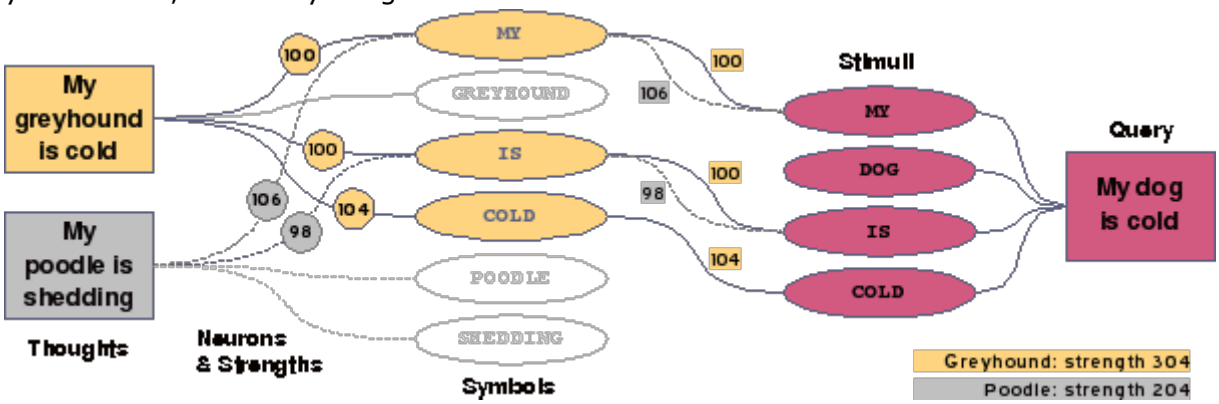
A *neuron* links a single thought with a single symbol. Each neuron has a strength that describe how relevant the symbol is to that particular thought. Each thought and each symbol can have many neurons.

Searching for Answers

What happens when you submit **"My dog is cold"** to System?

First, System breaks the question into search symbols, or *stimuli.*

Next, System identifies each symbol in its database that matches one of the stimuli, and lists the neurons linking those symbols to thoughts. For every thought returned, the strength of its neurons that took part in the search is summed, resulting in the final weight of that thought. You are presented with a list of the top thoughts that matched your search, sorted by weight.



Finally, you view the answer and provide feedback. Here's what happens when you click the "YES" (this answer was helpful) button:

Each neuron that in the viewed answer that took part in the search gets positively reinforced, increasing its strength.

> **Strengthened neurons for this thought:** MY IS COLD

Each neuron in that answer that didn't take part in the search gets negatively reinforced slightly, decreasing its strength.

> **Weakened neurons for this thought:** GREYHOUND

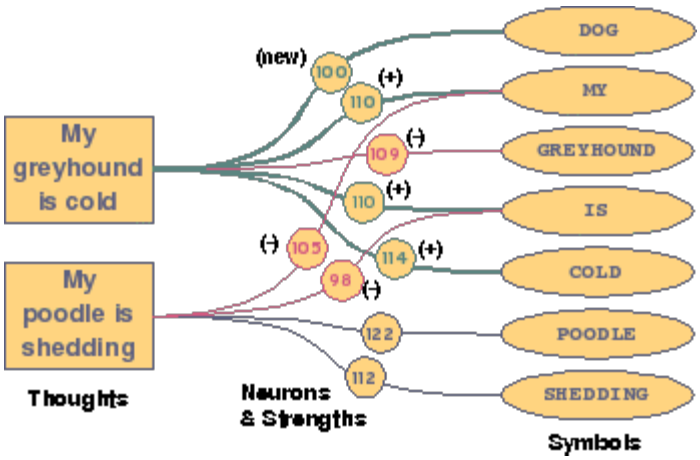
Each neuron that took part in the search but isn't attached to the selected answer gets negatively reinforced slightly, decreasing its strength.

> **Weakened neurons for other thoughts:** MY IS

Each stimulus that doesn't exist as a symbol gets added as a new symbol and gets linked to the selected answer.

> **New Symbols added and linked to this thought:** DOG

System has learned that DOG is relevant to this thought.



But what happens when you select the "NO" (this answer wasn't helpful) button?

Each neuron that in the viewed answer that took part in the search gets negatively reinforced slightly, decreasing its strength.

Weakened neurons for this thought: MY IS COLD

What's it all Mean?

First of all, it means System can learn new words and relate them to answers dynamically. Further, it can determine the importance of a given word with respect to a specific answer. It adapts. It is Borg.

Over time, System can identify common synonyms and mispellings automatically. This follows from the previous point; if you misspell a word, but eventually get to the right answer, System will add the misspelled word as a symbol.

Frequently-used answers bubble to the top, infrequently-used answers require more specific searches to find. Just like your own memory.

Noisewords and All That

What about "MY" and "HOW DO I" and all those other generally meaningless words that are getting strengthened during the searches?

System uses basic statistical analysis to identify words that occur so frequently as to be meaningless in a search. Words such as "A," "MY," etc., exist as symbols, but are generally ignored during a search. Note: The noise filter will not begin marking noisewords until there's enough data (from answers*and* searches) in your knowledgebase to be statistically significant.

Structure

Directory Structure:

ROOT :

- -adodb - adobdb framework classes
- Cache - cache folder 755 access required
- Classes - Util classes for engine
- Commands - Command classes , such as extract command , forget command , “mean” command to assign new meaning of the word

  Include - Main engine classes such as Mind Symbols ,

- Plugins - Engine Plugins
- Rdbms - Rdbms mapper
- api.php - Simple api return output in json format request/ learn methods
- Bootstrap.php - Main Bootstrap file (Must be included first in your project)
- Defines.php - Main Defines . path database etc…..
- Index.php - Simple interface for iteracting with the system .

Quick Start :

1 . Create Database and Import sql Dump

2. Edit Defined.php

3. navigate to index.php or include Bootstrap.php into your project and use methods listed below

Ask($string) - pass string for searching

Learng($string) - Learn new Knowledge

_____

Search, Learn, Commands

Searching the info :

1 User input phrase in natural language.

System break string into words and interpretative them as Word Objects with list of synonyms

So each word can has several meaning aka synonyms.

2 Learning input in your natural language

3 Commands

In Commands Folder there are several classes that can help you to create more flexible knowledge system.

You can use Specified commands after each answer . for correcting results . v

List of commands :

Bad_command : if the last answer was not acceptable enter keyword “wrong” it will decrease weights of last thought

Extract_command: Extract specified part of the last answer . For example if the answer was “The color is blue” you can enter “extract: blue” so the next time you ask about color will just outpurt “blue” note that there must be space beetweeb extact: and words list

Forget_command: forget last answer . keyword “forget”

Mean_command: Assign new Meaning-Synonims to Word . Example word1 mean word2

So in request stack word2 will be equal to word1

THankyou_command : Increase last answer weight . keyword ”thank you”

Examples:

For Examples :

If we gonna learn text “The sky is blue” , the system will break into neurons (aka words) and create knowledge by adding links between neurons.

So if we will ask “What color is the sky?” it will output “The sky is blue”, However maybe we want to get more accurate output lets say we want to get just name of the color . in this case we will need to use extract command.

1 Enter question “What color is the sky?”

2 enter “extract: blue”

3 That’s it .

Now if we ask the same question answer will be shorten “blue”.

The system uses predefined weights between neurons , so it is good practice to leave feedback to the system if the answer was correct, or answer was wrong.

For example after correct answer just enter “thank you” and if the answer was not informative enter “wrong”.

Happy codding. pashkovdenis@gmail.com