Which Features Matter How Much When?

Lee Scheffler
Cognitivity, U.S.A.
www.cognitivity.technology
lee@cognitivity.technology

Abstract

How do brains learn which features matter how much, when and for what purposes? A specific feature may matter more or less for recognitions of different learned patterns, and in different contexts and attentional foci. Simple executable "neural circuits" built from biologically-inspired reusable memory pattern components in the NeurOS™ and NeuroBlocks™ technology[1] model and implement a range of learning and dynamic contextual/situational/attentional feature relevance. A pattern is a collection of weighted features, roughly analogous to a neuron or neuron assembly. New patterns are created for sufficiently novel feature combinations. Individual feature weights in best-matching existing patterns grow or diminish with repetition, yielding patterns that adjust to repeated experience. Arbitrarily complex classification meshes typical of human knowledge are easily assembled by varying a simple novelty parameter. Cascading pattern recognitions build up layers of concrete to abstract feature vocabularies. Names or labels are modeled as synonyms for experience patterns. Context can be modeled as yet another feature, derived from recent activity, to discriminate among otherwise similar patterns. Attention can be modeled as broad dynamic parameters modulating feature signal strengths.

Keywords: features, labels, learning, memory, cognition, context, attention, cognitive simulation

1 Introduction

An abiding challenge in understanding how biological brains work, and in creating similarly intelligent machines, is in understanding representations and algorithms of the information processing involved (Marr, 1982 pp. 19-27). How do biological brains achieve complex cognitive capabilities by interconnecting a relatively small variety of building blocks: neurons of several types, synapses and dendrites. How does useful learning start from a small number of data points and continually adjust with experience? How do different personal life experience histories yield similar-enough knowledge frameworks?

[1] patents pending; see www.cognitivity.technology

This work follows Braitenberg's evolution-like "downhill invention" approach (Braitenberg, 1984 pp. 20-21): synthesize and incrementally improve working (artificial) systems that exhibit familiar cognitive capabilities. From these systems we may learn about core information representations and algorithms and system architectures underlying cognition. Insofar as these systems are built by interconnecting reusable biologically-inspired computational components in biologically plausible structures, we may draw some insights as to how corresponding capabilities may operate in biological brains, with rapid design iteration speeding hypothesis testing. And we may make strides along the path of creating artificial systems with similar capabilities. Put differently, let's start with simple assemblies of simple biologically-inspired components and see how far we can get.

What's a feature? For purposes of this paper, a feature is any distinct time-varying neural signal representing any percept or concept at any concrete through abstract level. This work explores how patterns of recurring feature combinations are learned, adjusted, classified, labeled, recognized and combined, and how recognition of such patterns vary with current context and attentional focus. Simple "neural circuits", using the NeurOS™ and NeuroBlocks™ technology, direct signals analogous to neuron spiking rates among reusable components performing biologically analogous computations and information storage. Small synthetic data sequence examples demonstrate the operation and learning dynamics of the assembled systems; these data are not "crafted" in any way other than to suggest one of many possible plausible sequences an organism might experience.

How can brains and intelligent systems learn so easily, quickly and effectively, especially from a few or even single examples, often with minimal or no supervision or reinforcement. How do such systems continually adjust to new experiences and environmental shifts? The bulk of current machine learning technologies require very large data sets, manual labeling, extensive parameterization of designs, take a long time to train and validate, and once trained are relatively static. Humans, by contrast, learn quickly from new examples and continually adjust what they know to new experience and corrections.

First, a single NeuroBlock "Set pattern" module managing an open collection of neuron-like feature patterns performs unsupervised learning of arbitrary classes. Learning creates new patterns or adjusts previous patterns. Varying matching and novelty parameters enables modeling of finely tuned exemplar patterns through increasingly broad stereotype patterns. Such a pattern is analogous to a "proxytype" (Prinz, 2002), where a learned pattern itself may be used as a feature input to other patterns. Feature inputs to a pattern can span multiple modalities and abstraction levels, depending on module connections.

Feeding the same input feature space in parallel to multiple such memory pattern modules with different matching and novelty parameter settings yields a rich classification mesh of learned exemplars and stereotypes that produces differential matching strengths to on-going experience.

Previously learned patterns are recognizable from a subset of their features, and their remembered features can merge with current input features to predict or "fill in the blanks" of missing features. Cascading pattern recognitions as features of other patterns yields layers of feature alphabets and vocabularies.

Names, labels and associated similar patterns are modeled as synonyms, a variation on a Set pattern with any/OR matching semantics: activity on any member feature activates the whole synonym pattern. Reification of such synonym patterns activates all synonyms, giving rapid access to similar patterns and labels.

Other cognitive processes (e.g., language recognition, other associational processes) may yield activation of a label, and, through reification of synonym patterns for the label, activate features associated with synonym patterns for the label, yielding a mechanism for imagination.

Interpretation of current inputs may depend on context. A word may have multiple meanings. The current context of a conversation, computed perhaps from recent words in a conversation, may disambiguate the word's meaning, elevating the activation of one candidate meaning over others.

Finally, attention is modeled as different dynamically adjustable multiplicative gains on feature signals. Such gains are computed from global parameters, which values in turn may be computed from other neural activity, modeling broad regional neuro-chemical influences.

A major point to all of these example learning and recognition neural circuits is that they are built largely using multiple instances of one common mechanism: a single Set pattern memory module. Simple parameter adjustments and interconnections exhibit the range of cognitive phenomena modeled.

Broadly, this paper can be viewed from two complementary viewpoints. The breadth of cognitive functions built bolster confidence in the strength and generality of the NeurOS cognitive systems architecture, the built-in NeuroBlock long-term memory pattern modules, and the rapid assembly and testing of cognitive architecture hypotheses enabled. The specific neural circuits demonstrated are plausible constructs to address important representation and algorithm challenges in cognitive research and engineering.

2 Materials and Methods

The cognitive functions modeled here are built and run using the NeurOS technology, one mission of which is to accelerate cognitive systems research and development. NeurOS enables rapid iterative assembly, running and instrumenting of cognitive functions by interconnecting reusable biologically-inspired component modules. See (Scheffler, 2015) and (Scheffler, 2016) for detailed descriptions of the technology and many usages. An overview sufficient to understand this work is provided here.

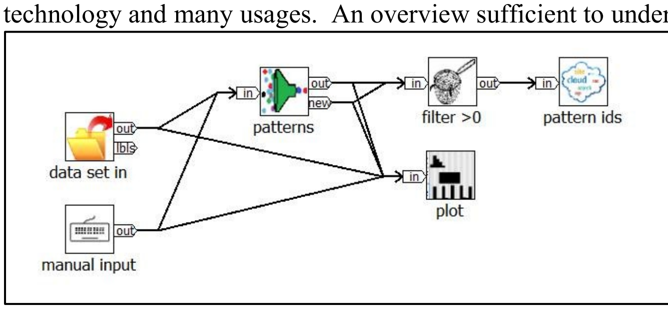

Figure 1: Simple neural circuit

Figure 1 shows a simple NeurOS **neural circuit** typical of this technology, built graphically using the NeurOS Designer tool. Iconic boxes are component **modules** called NeuroBlocks, each with function-specific input and output **ports**. Arrows are **links**, each conveying multiple feature signals from an output port of one module to an input port of a module, forming a directed graph, with loops allowed (not shown). (Details of this specific neural circuit are discussed below in Simple Unsupervised Learning and Recognition). Such neural circuits are directly executable by the portable NeurOS runtime system.

A module performs functions typical of a group or layer of multiple similar neurons or neuron assemblies, obtaining external inputs or feature signals on its input port(s), performing computation, updating internal state, and delivering external outputs or feature signals on its output port(s). A feature is encoded as a time-varying signal with a scalar value typically in the range (0,1) representing a fraction of a neuron's maximum output spiking rate[2]. A link conveys multiple such feature signals from one module to another, analogous to a "nerve" (axon bundle). Individual modules have function-

[2] This feature encoding, a single scalar value of a spiking rate over time, is the primary mechanism available to communicate among neurons. Any characteristic we might loosely call a feature (e.g., color=blue, size=large, duration=125msec) must be quantized in one or more such scalar feature signals (e.g., distinct R, G and B intensities, strengths of different{small, medium, large} signals, strengths of time interval quanta {10, 25, 50, 100, 200}.

specific **parameters**, analogous perhaps to different neuron/dendrite types and geometries. Some module parameters can vary dynamically during circuit operation by referring to variable **global parameters**, typically modeling broad neuro-chemical signaling effects.

NeurOS includes many built-in module types, as well as facilities to easily incorporate external programs or write custom modules. Individual module functions are described as used in this paper.

Several **long-term memory pattern modules** learn and recognize spatio-temporal combinations of input signals, similar to *proxytypes* (Prinz, 2002). A **Set module**, used heavily here, learns and recognizes sets of concurrently active (non-0) feature signals. One Set pattern is analogous to a neuron that learns a pattern via adjustments to conductivities of its input synapses. A similar **Sequence module** learns and recognizes sequential combinations of feature signal changes without regard to time, and a Temporal module learns and recognizes combinations of feature signal changes including relative time intervals. A complementary **Reify module** regenerates signals for all the component features of a pattern, and is often used for prediction and imagination. These components model an implicit sparse coding of a feature space.

A memory pattern exists in one **pattern space**. Feature pattern novelty is judged within one space. Unlike conventional neural networks, the input feature space of patterns in one pattern space is unbounded, as is the number of distinct patterns. New features created (e.g., by learning new patterns) can flow over existing links with no circuit change, and can participate in subsequent pattern learning and recognition. Multiple pattern spaces model distinct physical or functional brain regions.

Simple Set patterns used here follow a classical neuron mathematical model: a non-0 numeric weight associated with each input feature signal that affects the pattern. Set pattern matching strength is computed as a familiar sum of products of feature signal strengths and corresponding weights, scaled by a parameterized normalizing curve that determines the semantics of auto-associative matching, over a range spanning {any/OR, a few, some, many, most, all/AND}[3].

Learning is local to one pattern space and follows simple rules, similar to classical nearest-neighbor cluster learning:

1. If a current feature combination matches an existing pattern well enough (matching strength exceeds a threshold parameter), the feature weights of the best matching pattern(s) are adjusted toward current signal strengths according to the pattern module's learning rate parameter and matching history, possibly adding or removing features from the pattern as needed.
2. If no existing pattern matches a current feature combination well enough, a new pattern is created with feature weights based on current signal strengths, analogous to recruiting a new neuron to recognize the novel pattern. (This is akin to dynamically adding a new neuron-like unit as needed to a conventional artificial neural network layer.)

A single Set pattern is thus akin to a cluster in feature space. One Set module and its associated pattern space effectively defines a classical single layer neural network, with the addition that the population of inputs and the number of units is unbounded. No backtracking or static training is involved. Learning is continuous, adjusting pattern populations and feature weights with continued experience. Cross-inhibition among patterns is not directly represented, but emerges from post-processing of activation strengths among multiple recognition candidates.

Sequence and temporal pattern matching and learning operate similarly; see (Scheffler, 2014) for details.

The current work explores how far we can get using just these built-in memory module and functional module types. Additional memory models and functionality can be easily added with

[3] Additional pattern capabilities, not discussed here, include expected-value distributions for each feature (Kurzweil, 2012 p. 48), error/noise tolerance and salience/permanence for forgetting.

NeurOS customization features. Complex capabilities are created by interconnecting multiples of these modules, reminiscent of convolutional and recurrent neural networks.

Because learning in this architecture works with relatively few examples, and because human learning progresses continuously from very few initial experiences, very small synthetic training data sequences were used.

For presentation clarity, feature signals discussed here are often assigned symbolic and contextually relevant semantic IDs, for example "barks" or "word_3" instead of #1d2e3f; in a practical system such signals would have no more identification than individual neurons/axons in a biological brain.

3 Results

A progression of NeurOS neural circuits were built that exhibit a variety of feature pattern learning and recognition. These circuits show how feature strengths in patterns change with experience, and how pattern recognitions change dynamically with context and attention.

3.1 Simple Unsupervised Learning and Recognition

The neural circuit in Figure 1 operates as follows:

- The "data set in" module reads from a file, each line representing a data point of concurrent feature identifiers. For each feature identifier, it emits a signal with a value of 1 (maximum spiking rate) for 40 msec. Labels associated with data points are ignored in this usage, indicated by the "lbls" output port remaining unconnected.
- The "patterns" module is a Set pattern recognition and learning module. It tries to match each new concurrently active feature set with existing patterns in its pattern space, and either adjusts feature weights of the best matching pattern(s) or creates a new pattern for sufficiently novel feature combinations. It sends a feature signal for each matched pattern on its "out" port, with a value reflecting pattern matching confidence. It emits a feature signal with a value of 1 on its "new" output port each time a new pattern is created.
- Additional modules provide human-readable output. The "filter >0" module feeding the "pattern ids" module yields a "cloud" of pattern identifiers as they are matched, with font size proportional to matching strength. The "plot" module shows signal strengths of input features and recognized/new patterns in an EKG-like display over time.
- The "manual input" keyboard module allows hand-entry of data points as lists of feature identifiers, for exploration and testing.

Figure 2 shows possible parameter settings of the "patterns" Set pattern module. Patterns live in a memory pattern space named "exemplars" in this case. Learning is enabled. The learning rate controls how much newly arriving features affect the weights of the best matching pattern: higher values favor new data over past history. A new pattern is created only for a minimum of 4 concurrently active input features. (Fewer features may still participate in pattern matching, just not new pattern creation.) The response curve indicates that a minimal match score requires at least 25% of the feature-weight products, and grows linearly to a match score of 1 when 100% of the feature-weight products are present, yielding a "many-most" pattern matching semantic. The new pattern (novelty) threshold is roughly the "cluster narrowness" of each pattern in "feature space". High values (as in this case) yield many highly specific narrow exemplar patterns, while lower values yield fewer broader stereotype patterns.

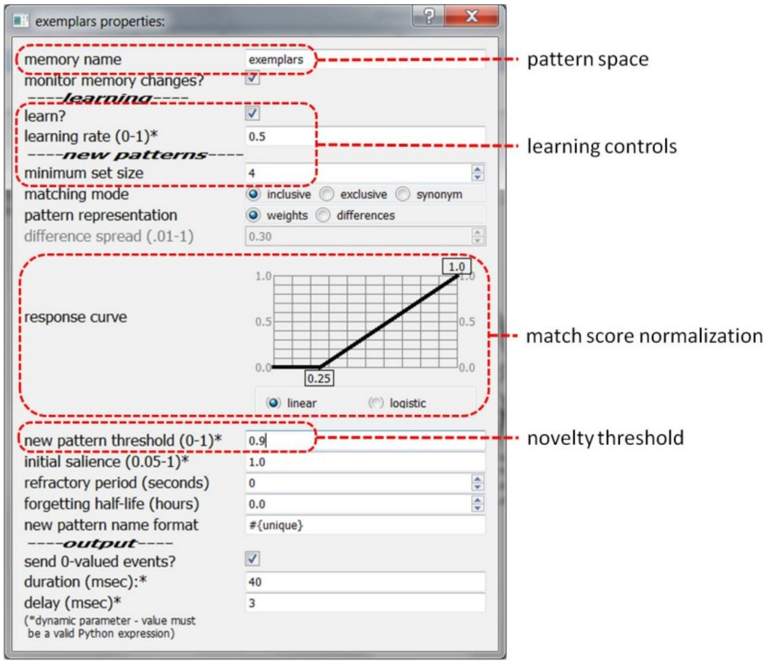

Figure 2: Set pattern module parameters

Using the simple data set below, we re-run the neural circuit of Figure 1, varying the response curve and novelty parameters. The data set is in a simple "label = features" file format. Labels are ignored in this first usage, indicated by the "lbls" port on the "data set in" module remaining unconnected. For simplicity in this example, all features have a(n implicit) value/strength of 1.

```
dog = fur, snout, barks, big, 4 feet, brown, tail
dog = hair, snout, whines, medium, 4 feet, black, tail, friendly, slobbers
dog = fur, yips, small, 4 feet, white, nasty
cat = fur, tail, small, black, 4 feet, pointy ears, meows, whiskers
cat = fur, tail, small, grey, 4 feet, pointy ears, meows, whiskers
cat = tail, calico, purrs, 4 feet, whiskers, claws
```

Table 1 shows pattern feature weights from running the same neural circuit from scratch (no memory between runs) with several different parameter settings. Blank table cells represent either a) feature is not part of the pattern (equivalent of 0 feature weight), or b) after learning the feature weight diminished below a significance threshold of 0.1 and was eliminated from the pattern.

- The columns labeled #1 through #6 show individual exemplar patterns learned for this data set, courtesy of a "many-most" response curve semantic and a high novelty threshold: each new pattern was sufficiently different from any previous one to cause creating a new narrow exemplar pattern.
- The columns subgroup_1 and subgroup_2 show feature weights of two distinct stereotype patterns resulting from a moderate novelty threshold and a "some" response curve semantic. Careful examination of the input data and the weights table reveals that, while the first two "dog" data points combined into the "subgroup_1" pattern, the third "dog" data point was sufficiently different that the new "subgroup_2" pattern was created; subsequently the several "cat" data points matched this cluster better than subgroup_1. This is to be expected: this is ***unsupervised*** learning, so only feature similarity matters and labels categorizing the data points are not part of the learning.

- Finally the "group_1" column results from a low novelty threshold and a "some" matching response curve semantic. This single stereotype records repeatedly updated commingled feature weights for all the data points. It shows clear evidence of preference for the features of later data points over earlier ones, courtesy of the high learning rate. Clearly color and personality matter much less than having 4 feet, fur and a tail for membership in this group.

novelty:	*0.9*						*0.59*		*0.5*
curve min-max:	*0.25 - 1*						*0 - 0.5*		*0 - 0.3*
pattern id:	#1	#2	#3	#4	#5	#6	subgroup_1	subgroup_2	group_1
4 feet	1	1	1	1	1	1	1	1	1
barks	1						0.69		0.14
big	1						0.69		0.14
black		1		1			0.31	0.18	0.22
brown	1						0.69		0.14
calico						1		0.23	0.33
claws						1		0.23	0.33
friendly		1					0.31		
fur	1		1	1	1	1	0.69	0.77	0.59
grey					1			0.26	0.22
hair		1					0.31		
medium		1					0.31		
meows				1	1			0.43	0.37
nasty			1					0.33	
pointy ears				1	1			0.43	0.37
purrs						1		0.23	0.33
slobbers		1					0.31		
small			1	1	1			0.77	0.45
snout	1	1					1		0.22
tail	1	1		1	1	1	1	0.67	0.92
whines		1					0.31		
whiskers				1	1	1		0.67	0.7
white			1					0.33	
yips			1					0.33	

Table 1: Unsupervised pattern learning

The weights in Table 1 result from just a single pass over the data set in the order shown, and therefore exhibit order-based biases. Rerunning the same data set multiple times causes these weights to settle down to relatively stable values (not shown here). Running the same data points in different/random orders can yield modestly different stereotype subgroups. This phenomenon was not studied further for this paper; see Future Work.

After this (brief) training, subsequent manual input of partial feature combinations yields multiple matching scores for each of the learned patterns, as shown in Table 2.

curve min-max:	*0.25 - 1*						*0 - 0.5*		*0 - 0.3*
pattern id:	#1	#2	#3	#4	#5	#6	subgroup_1	subgroup_2	group_1
fur, barks, tail	0.24						0.62	0.42	0.85
small, meows, whiskers				0.17	0.17			0.54	0.78
small, meows, whiskers, black				0.33	0.17		0.08	0.60	0.9
big, brown, barks, tail	0.43						0.8	0.19	0.69

Table 2: Example pattern recognition scores

3.2 Concurrent Classification and Recognition

Human classification tends to be messy and only quasi-hierarchical. A new input data item (set of active features) can concurrently be all of Fido, a golden retriever, a dog, an animal, a pet, a friend, a service animal, a road hazard. Recognitions are often differential, with several possibilities activated with different strengths, especially in the face of partial information.

This suggests a neural circuit like that in Figure 3 *(a)*.

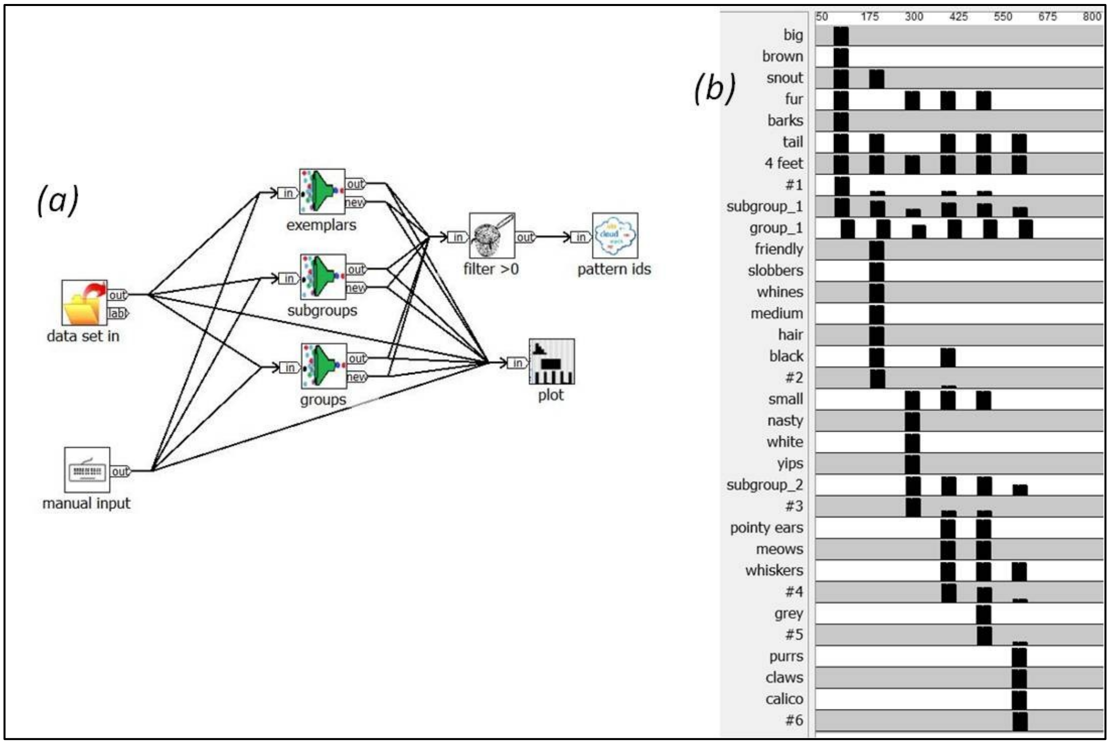

Figure 3: Concurrent multi-classification

Three distinct Set pattern modules, "exemplars", "subgroups" and "groups" learn and recognize concurrent input feature combinations in parallel, each with different novelty and response curve parameters as in Table 1, and producing the same concurrent pattern recognition scores as in Table 2. Figure 3 *(b)* shows the progress of feature arrival and concurrent pattern recognition and creation over time. All the exemplar and stereotype patterns are available for recognition and additional learning for subsequent inputs.

Note that just two neuron-equivalent patterns (subgroup_1 and subgrooup_2) are needed to represent essential feature similarities/differences among data points, and an additional one (group_1) to capture commonalities and relative feature contributions for all patterns.

3.3 Prediction

Learned patterns make it possible to predict features not actually experienced and even correct perceptual mistakes and uncertainties.

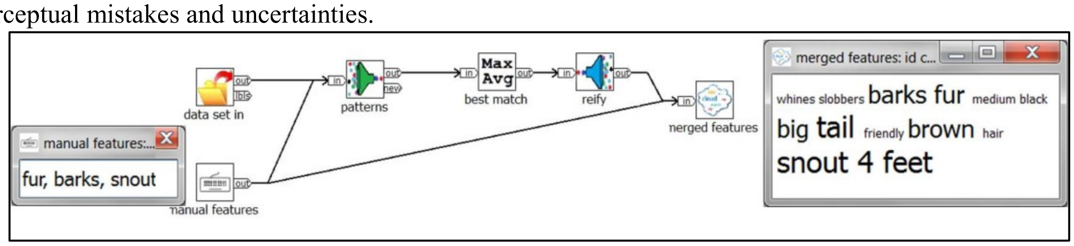

Figure 4: Prediction

In Figure 4, after the initial learning from data (the "data set in" module), manual input of a subset of features via the "manual features" keyboard input module leads to pattern recognition in the "patterns" module, yielding one or several pattern matching signal events with associated matching scores. The "best match" module picks the pattern with the highest current matching score, and the "reify" module regenerates signals for all the previously learned features of that pattern, with signal strengths proportional to both the pattern recognition score and the weight of the feature within that pattern. These feature signals are commingled at the input to the "merged features" id cloud module, yielding a composite of both experienced and predicted features, with font size indicating relative feature strengths. Observing just "fur", "barks" and "snout" predicts "4 feet" and "tail" strongly, likely "big" and "brown", with other features possible but with low likelihood.

3.4 Layers

Simple pattern learning and matching serves as a repeated building block for higher levels of cognition. If we cascade two or more successive pattern modules, with recognized patterns at one layer as input features to the next layer, as shown in Figure 5 *(a)*, we achieve an adaptive hierarchy of what we might call "alphabets" or "vocabularies". Visual edges/curves/corners become letters, letters become words, words become phrases. Edges/lines/corners become shapes become objects become scenes. With continued experience, familiar combinations at one level emerge as the most likely interpretations of continued input features. As one vocabulary level settles/converges on the most common patterns at its level, these patterns as features lead to progressive settling/convergence at successively higher conceptual levels. Intuitively, this is reminiscent of childhood learning. Complex experience patterns at one "level" remain disorganized until some mastery (≈ pattern recognition stability) of basics at lower levels is achieved. Thereafter these features become stable contributors to next layer(s) of patterns.

In neural circuit *(a)* the "LED segments" file reading module simulates successive observations of a typical 14-segment digital LED display *(b)*, each file line yielding concurrent signal bursts for the on (lit) segments of the display for one letter observation. The "letter patterns" Set module recognizes and emits signal bursts for one (or possibly more) candidate letters; the "max letter" module passes along just the strongest candidate. "words" is a Sequence pattern module that recognizes sequences of input signals, in this case individual words; "phrases" is a similar Sequence module. Both modules share previously learned patterns in the same memory space, with one relevant sequence pattern shown in *(c)*. The EKG-like signal plot *(d)* shows the action over time. Signal lines are sorted in first non-0 signal appearance order. (The "add prefix" module simply prepends "segment_" to signal IDs for display clarity.)

In particular, as enough letters accumulate in sequence to spell the word "four", slight recognition of the gettysburg_address" phrase begins. Recognition of additional words increases phrase recognition progressively as highlighted in red.

Those familiar with today's "deep learning" machine learning approaches, particularly convolutional neural nets (CNNs), will no doubt recognize similarities. See Summary and Discussion for differences and continuous incremental learning.

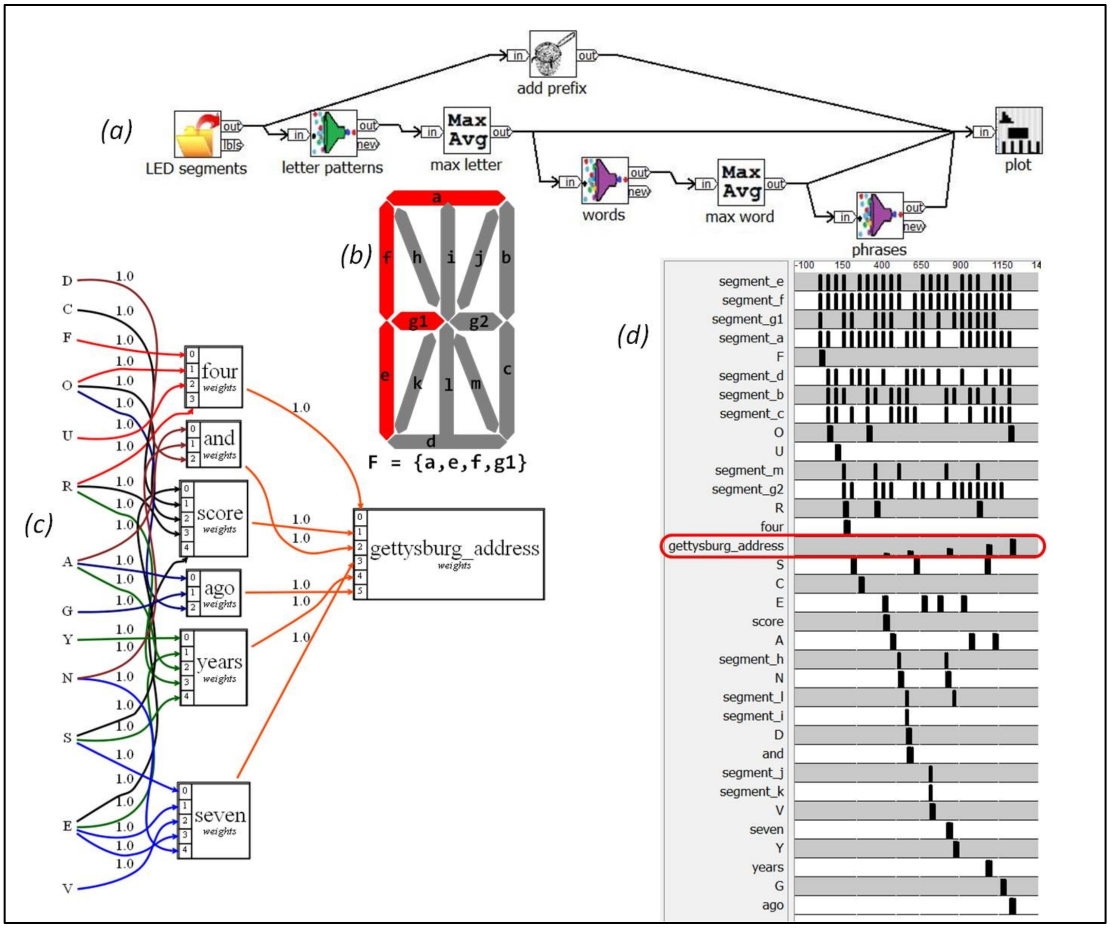

Figure 5: Layers of patterns

3.5 Labels as synonyms

This section explores hypotheses of labels as synonyms for patterns. In this view, there is nothing special about a label: it is "just another feature" associated with a combination(s) of other features.

A second cascaded Set pattern module creates patterns that associate labels with other feature patterns, using a "synonym" semantic where any activity on any input member to a pattern stimulates the pattern, similar to an OR function. Reification of the synonym pattern yields all its components: labels and associated patterns.

- A label may be associated with a feature pattern, like an exemplar name (e.g., "Fido") or a classical best category or class label (e.g., "dog", "animal", "living thing", "friend", etc.)
- Multiple labels, such as names, sounds or symbols, can be associated with a feature pattern, often a stereotype pattern (e.g., "dog").
- A label (e.g., "dog") may be associated with multiple feature patterns, often exemplars (e.g., multiple specific dogs) or multiple stereotypes.
- Multiple patterns may be synonyms of each other, as in the case of multiple "views" of the same or similar objects, including features across sensorimotor domains.

The neural circuit of Figure 6 demonstrates learning these kinds of relationships.

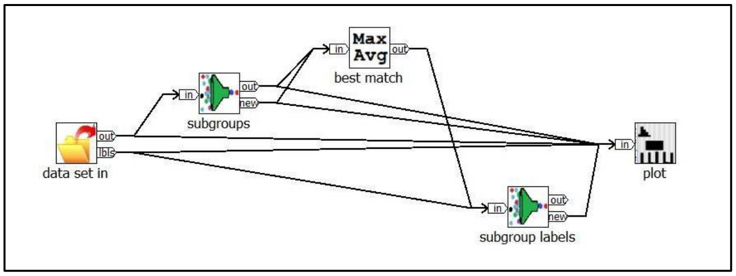

Figure 6: Assigning labels to patterns

The previous data set is used, this time with the "lbls" output connected. The parameters of the "subgroups" module yield three distinct subgroups from the data set, as shown in Table 3 below. The label "dog" is associated with subgroups 1 and 2, while the label "cat" is associated with subgroups 2 and 3. Feeding the label "dog" to the "subgroup labels" Set pattern module yields the "subgroup_labels_1" synonym pattern which, when reified, yields signals for both "subgroup_1" and "subgroup_2" patterns, modulated by their respective weights (0.6 and 0.33). Reifying each of these yields the combination of features in both subgroup patterns.

data points	*patterns/labels*	*subgroup_labels_1*	*subgroup_labels_2*
dogs 1, 2	subgroup_1	0.6	
	dog	1.0	
dog 3, cats 4, 5	subgroup_2	0.33	0.59
cat 6	subgroup_3		0.33
	cat		1.0

Table 3: Label synonyms

Variations of parameters can use this same neural circuit pattern to assign labels to individual exemplars (e.g., "Fido" to the first data point, etc.), or to the whole data set (e.g., "animal" to the overall group stereotype).

3.6 Imagination

A label can serve as a generator of a pattern of features. Hear or read "dog", or have higher-level cognitive processing generate the label "dog", and the features we have come to associate with dogs "spring to mind". Through the synonym patterns introduced above, activation of a label feature leads to activation of its synonym pattern(s). Reification of those patterns activates the pattern member features, "bringing to mind" all the features of the synonym-associated pattern(s), with strengths proportional to their weights in the pattern. So, for example, to imagine a "cat", enter the label "cat" in the "label input" module in Figure 7, activating features we have learned to associate with cats in our experience.

Figure 7 adds a new bottom path to the neural circuit of Figure 6**Error! Reference source not found.**. Entering a label like "cat" in the "label input" keyboard input module causes the "find synonyms" Set module to find synonym patterns including "cat". "reify_synonyms" regenerates all the synonyms for "cat" including possibly many patterns. "reify_features" subsequently regenerates all the features of all the synonym patterns for "cat", with individual feature strengths of the activated patterns commingled additively and displayed by the "features" word-cloud module with font size proportional to relative feature strength.

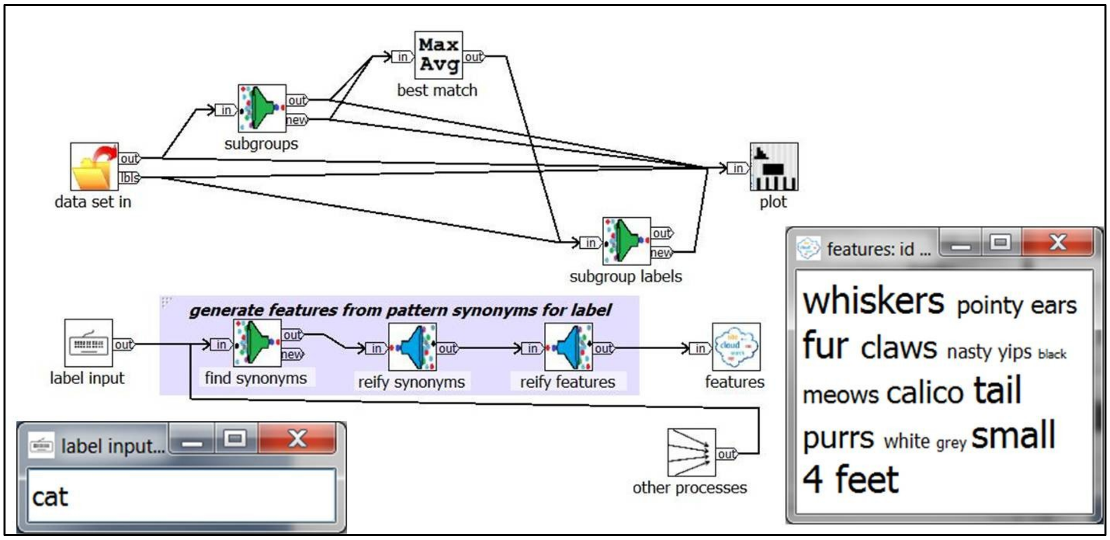

Figure 7: Imagination

Note that relative feature strengths indicated are as previously learned from the data (weighted by the most recent encounter), not necessarily what we would think of after our lifetime of adjustment of feature weights. The "other processes" module and path is a place-holder to indicate that the label input to this neural circuit might also come from other cognitive processes, such as language parsing or higher-level imagination.

3.7 Context

Our interpretations of ambiguous words like "beat", "pitch", and "batter" depend strongly on the context of our recent conversation, for example: music, sports or cooking. Conversely, the context of a conversation derives (at least in part) from words recently perceived (or imagined).

The neural circuit *(a)* in Figure 8 demonstrates mechanisms for simultaneously computing context and progressively adjusting context-dependent meanings of words.

The "words in" keyboard input module is configured to emit successive white-space-separated words as individual signal bursts (value=1, 40 msec duration). The following "keep keywords" Transformer module discards most non-contextual words (e.g., articles, prepositions and pronouns) purely to simplify results display. Three successive word sequences *(c)* are shown, all using the word "beat", which takes on different meanings in each sentence. Successive word features are persisted for a few seconds by the "working mem" module. These features are combined in the "context lookup" Set module to progressively establish possible current contexts (cooking_context, music_context, sports_context) based on previously learned terminology for each context *(b)*, with signal values reflecting confidence. The "meaning lookup" Set module recognizes patterns combining a current word and current contexts to generate possible meanings (previously learned), with corresponding signal strengths indicating confidence, as highlighted in the EKG-like combined "plot" module display *(d)*.

The word "beat" takes on different meanings depending on context: **green** for the **cooking_context**, **blue** for the **music_context**, and **purple** for the **sports_context**. Note that, although "beat" occurs early or mid-way in each sentence, its meaning strength(s) grows progressively as the current context is more firmly established with later words in each sentence.

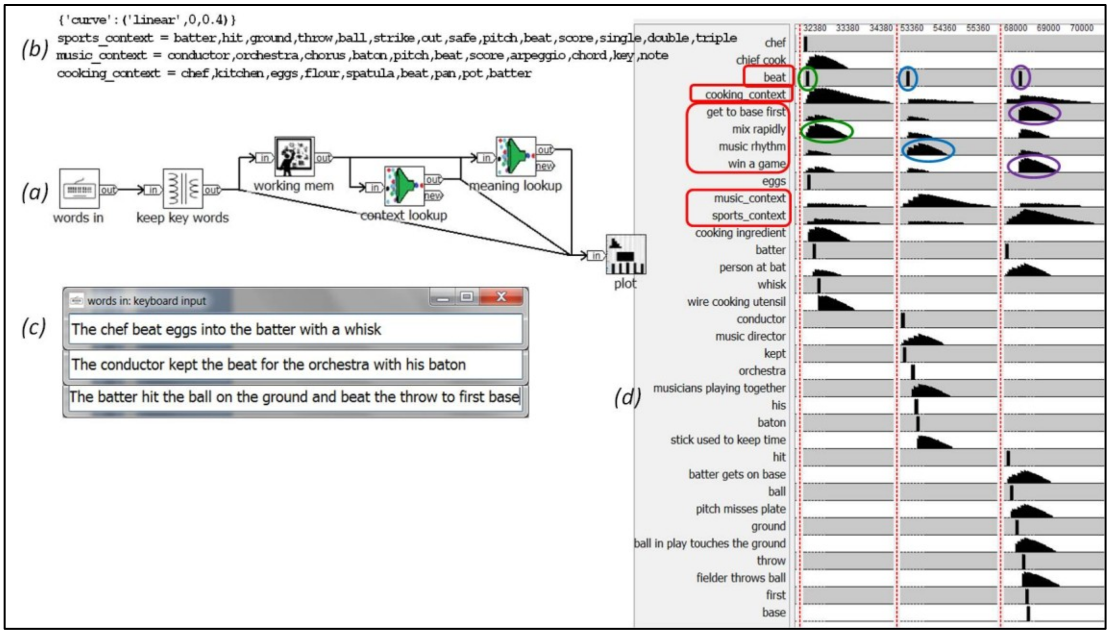

Figure 8: Context

3.8 Attention

Attention and focus are complex and multi-faceted. Conceptually, various biological mechanisms act to increase activation of features important at the moment (thereby enhancing downstream processing activity) and decrease activation of other features. In the limited space available here, we explore just one of many possible NeurOS usages to loosely model the gating effects of neuro-chemical concentrations on neural activity: shared dynamic parameters and feature signal gain parameters.

Figure 9 is a highly simplified model of how observation of potentially dangerous features may enhance attention to external perception.

In neural circuit *(a)* the "sense inputs" keyboard input module simulates a variety of observed sensory signals *(e)*[4], emitting them as full-strength (value=1) 40-msec-long signal bursts in sequence 100 msec apart. The "working mem" module persists and decays these signals exponentially with a half-life of 500 msec. The "danger patterns" Set pattern module is initialized with a pattern *(b)* of certain features that mean danger. (In a realistic system these patterns might be learned and have non-uniform feature weights proportional to "how much danger".) As features arrive, this module emits a signal for the "danger" pattern with strength proportional to the cumulative strengths of dangerous features currently observed. The "update DANGER" Transformer module sets the value of the graph-level shared parameter G.DANGER to the current strength of the "danger" pattern via expression *(c)*. Along the bottom path, the "apply gain" Filter module uses expression *(d)* to modulate feature signal strengths. When there is no current danger (G.DANGER is 0) then the perceptual input signal gain is 0.5, perhaps corresponding to normal background environmental awareness. As features arrive that elevate G.DANGER (e.g. "tiger"), the gain increases and incoming raw sensory signals propagate with increased values, loosely modeling biological neuro-chemical effects on synapses. As time passes with no new activity from potentially dangerous incoming features, G.DANGER decays and the net modulated strengths of incoming perceptual feature signals return to their "normal" levels.

---

[4] As in other examples, these signals are named symbolically for explanation clarity. In a realistic system these are simply signals derived from perceptual processes with non-symbolic identifiers.

All this is shown graphically in *(f)*. The initial tree, grass and tiger signals show a "normal" gain of 0.5. The first recognition of "tiger" stimulates "danger", which in turn enhances the strengths of all subsequently arriving perceptual signals. Subsequent perceptions of "roaring" and "snake" further boost the gain of external perceptual signals. As time passes with no new dangerous feature perceptions, "danger" decays and the gains on perceptual features (e.g., repeated perceptions of "grass" shown) settle to their normal values.

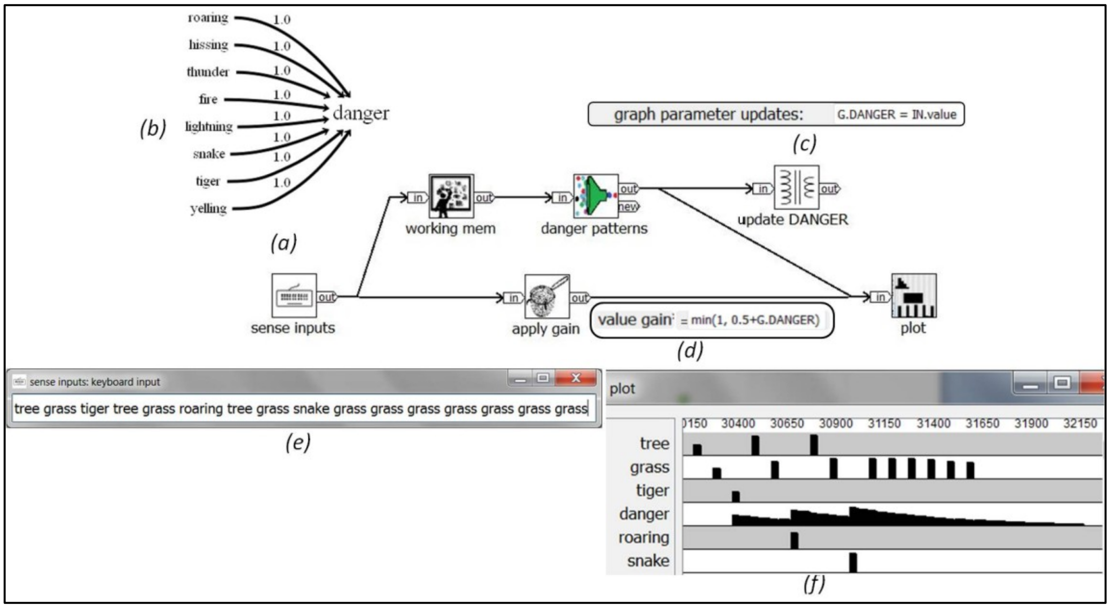

Figure 9: Attention

This same machinery can of course have other effects on other neural signal paths. For example, a different gain expression such as "1-G.DANGER" might modulate imagination or "thinking" feature signal paths, modeling removing attention from those paths in the face of danger. To model sleeping or exhaustion, where perceptual gain is further turned down, the expression *(d)* could further include other graph-level shared parameters modeling these mental "states".

4 Summary and Discussion

This work has explored plausible biologically-inspired mechanisms for how features at various levels of abstraction can combine and recombine into structures of long-term memory patterns, and how feature-based pattern recognition may vary with context and attention. Small NeurOS "neural circuits" were built by interconnecting NeuroBlock modules for inputs, processing, outputs, and in particular memory pattern modules. Variations in simple parameters and signal interconnections of biologically-inspired Set and Sequence pattern modules yielded a wide range of cognitive capabilities: concurrent parallel unsupervised learning of exemplar and stereotype patterns, prediction of missing features, layers of alphabet/vocabulary patterns, labels as synonyms, imagination expansion of labels into features, and variations in pattern recognition strengths from the same features in different contexts and attentional states.

Categorization is coming to be viewed not as a distinct task but as part of the overall cognitive processing fabric (Anderson, et al., 2001). The neural circuits here can be replicated anywhere in a cognitive architecture where such pattern learning and recognition is needed.

Concept learning and recognition systems benefit from concurrent hybrids of exemplar, stereotype/prototype and other categorization systems. (Lieto, 2014) (Lieto, et al., 2015) Small neural circuit sub-assemblies of NeurOS memory pattern modules, including the NeurOS Set pattern "synonym" semantic, enable creation of such unified hybrid concept representations.

One cognitive architectural viewpoint emerging from this work is that low-level features continue to matter the same amount (same weights) as learned from repeated experience by low-level patterns, but that higher-level "upstream" processes involving attention, context and recombinations with other features gate the current importance of a lower-level feature. A yellow car may not mean much unless you are looking for a taxi in New York City. Such a suggestion is biologically testable (e.g., brain activity scans) and is an example of potential bi-directional synergy between biological and artificial cognitive research.

The fact that these diverse cognitive constructs emerged from combining relatively few common building blocks builds confidence both in sufficiency of the technology, and in continuing such synthetic approaches. These low-level cognitive sub-assemblies may themselves be reused and combined to build increasingly complex cognitive systems.

Parametric variations of the Set and Sequence modules yielding different pattern learning and recognition semantics are suggestive of functional differences among biological neuron types. Shared global parameter variables seem reasonably model broad neuro-chemical signaling paths. Reification of a learned pattern into activation of its component features seems powerful and essential, and may be analogous to extensive feedback connections found in biological brains.

Biological brains exhibit a rather large number of inhibitory synapses (Seung, 2013 p. 56), particularly those used in cross-inhibition where the strongest neuron activation (highest spiking rate) decreases firing of other competing neurons. This work so far does not explicitly model negative-weighted inhibitory synapses, although NeurOS machinery can handle them (as negative feature weights in patterns). Rather, the functional effects of cross-inhibition emerge in these neural circuits from using a max function in a Group Operations module to select and pass through the strongest input signal of many.

Examples and data used here are simplified in numerous ways. In more realistic systems noise and sensor characteristics will yield less ideal input feature signals. NeurOS easily handles such variable signals, but they are not shown here to simplify presentation and explanation.

Learning here diverges from current popular machine learning approaches (Domingos, 2015 p. 7). One-shot learning from a single feature data point is immediately available for subsequent recognition. Learning is continuous with on-going experience. There are no inherent limits on the number of patterns that can be learned. Instead, repeated similarities implicitly bound the pattern population.

Not surprisingly, what is learned is highly order dependent. The need for order-independent static model convergence and stability are questionable. Human classification is inherently messy and subject to progressive adjustment from experience. Convergence of pattern layers despite such order-dependence is explored in Future Work.

Backtracking is not used for supervised learning. Instead, labels are associated with learned patterns as synonyms in a feed-forward fashion.

The history of artificial intelligence covers the gamut from highly abstract "symbolic AI" to exceptionally detailed biological simulation models of neurons, synapses, dendrites and axons. NeurOS focuses on function and connectivity while maintaining plausible analogies to biology, offering numerous benefits. Working neural circuit models are visual, resembling functional and brain connectivity flow diagrams, and are computationally tractable, easy to test, instrument and rapidly iterate. Neural/cognitive research findings can be rapidly implemented and tested, and easily embodied in artificial systems. Biological function hypotheses can be quickly simulated, tested, instrumented, iterated and understood in ways that are exceedingly difficult in vitro. This encourages synergistic cross-inspiration between biological and cognitive computing research and development.

4.1 Future Work

These and other neural circuits (see (Scheffler, 2014) and (Scheffler, 2016)) that have been built are just starting points of what promises to be a rich journey of understanding. On-going work is anticipated along many vectors to build increasingly diverse, complex and capable working cognitive systems.

Adding new NeuroBlock modules to encapsulate perception and action technologies will increase the scope and range of cognitive systems that can be built, and enable them to interface increasingly with "real worlds", including not just the physical world (e.g., in robotics), but also the electronic worlds of the Internet and the "Internet of Things".

Set, Sequence and Temporal long-term memory pattern modules are broadly useful. Nevertheless, one size need not fit all. Biological brains use similar components and structures in many places, but also incorporate specialized structures and components. Using NeurOS facilities to integrate existing technologies for machine learning, vision and speech processing, robotic systems, planning, language understanding and production will enable richer and more competent artificial systems.

As neural circuits emerge for generally useful cognitive functions, they can be recaptured as reusable NeurOS modules in similar fashion to electronic integrated circuits, and shared with the NeurOS user community.

In the spirit of evolution and Braitenberg's "downhill invention" (Braitenberg, 1984), incremental improvements to the cognitive constructs reported here may yield increased capabilities and understanding:

- Replacing the 14-segment LED input with more general pixel-grid input and visual preprocessing may move the neural circuit of Figure 5**Error! Reference source not found.** closer to handwritten digit and character recognition.
- Adding a phonetic path to the neural circuit of Figure 5**Error! Reference source not found.** may test reading theories of commingling learned audio and visual alphabets and patterns.
- More capable reading and language understanding neural circuits may use multiple recombinant paths for letter alternatives, n-grams, syllables, short phrases, synonyms, etc.
- Adding feedback connections from prediction (e.g., next word possibilities from familiar phrases) may enhance reading speed.
- Having an existing pattern label available with new data may help focus classification adjustments for new data points: if you are told an animal is a cat, you can adjust just cat-synonym feature patterns.
- How can different brains with essentially the same representations and algorithms a) develop divergent knowledge infrastructures (alphabets and vocabularies), yet b) exhibit similar higher-level understanding and capability. Stereotype clusters learned as in Simple Unsupervised Learning above are clearly dependent on the order of experience. Higher layers of pattern learning and recognition recombine lower-layer alphabets and vocabularies. With an eye to algebraic equivalence of complex expressions, abstract levels of similar understanding may still emerge from combining different finer-grained vocabularies, in the mode of Layers above. Presenting similar but different experience data point sequences should yield differing mid-level patterns but convergent higher-level recognition results.
- Context may derive from more than just current conversation, but include situational awareness (e.g., at the ballpark, concert hall or in the kitchen) and "what I was thinking about".
- Additional attention mechanisms, such as generating behavioral actions to modulate sensory input (e.g., eye/head/eyelid movement) and modeling more complex neuro-chemical interactions.

Humans are equipped to do a "pretty good" job at most endeavors, although machines of focused optimized design exceed human performance at specific tasks. NeurOS-based cognitive systems are expected to pursue today's popular cognitive challenges: handwritten digit and letter recognition, scene segmentation, object recognition, written and spoken language understanding, optical illusion resolution. Like humans, a "general intelligence" should do well in a "Cognitive Olympics" of many diverse events, even if it does not beat all comers at any specific event.

A NeurOS/NeuroBlocks development kit and all the neural circuits and related data sets used here are expected to be available soon; see (Scheffler, 2016). An open community of NeurOS users and developers are expected to grow the technology and its usages in numerous dimensions: more modules, more sensorimotor functions, ports to more platforms, performance increases from both parallelism and modern neuromorphic hardware support, distributed and team operation.

References

Anderson J. R. and Betz J. A hybrid model of categorization [Journal] // Psychonomic Bulletin & Review. - 2001. - 4 : Vol. 8. - pp. 629-647.

Braitenberg Valentino VEHICLES - Experiments in Synthetic Psychology [Book]. - Cambridge : MIT Press, 1984.

Domingos Pedro The Master Algorithm [Book]. - New York : Basic Books, 2015.

Kurzweil Raymond How to Create a Mind [Book]. - USA : Viking Penguin, 2012.

Lieto A A computational framework for concept representation in cognitive systems and architectures: Concepts as heterogeneous proxytypes [Conference] // Procediat Computer Science, Proceedings of BICA 2014. - 2014. - Vol. 41. - pp. 6-14.

Lieto A., Radicioni D. P. and Rho V. A Common-Sense Conceptual Categorization System Integrating Heterogeneous Proxytypes and the Dual Process of Reasoning [Conference] // Proceedings of the Internaton Joint Converence on Artificail Intelligence (IJCAI). - Buenos Aires : AAAI press, 2015. - pp. 875-881.

Marr David VISION - A Computational Investigation into the Human Representation and Processing of Visual Information [Book]. - San Francisco : W. H. Freeman, 1982.

Prinz Jesse J. Furnishing the mind: Concepts and their perceptual basis [Book]. - [s.l.] : MIT Press, 2002.

Scheffler Lee Cognitivity [Online] // Cognitivity. - 2016. - http://www.cognitivity.technology.

Scheffler Lee NeurOS and NeuroBlocks - a neural/cognitive operating system and building blocks [Journal] // Biologically Inspired Cognitive Architectures. - Newton, MA : Elsevier, 2014. - January 2015 : Vol. 11. - pp. 75-105.

Scheffler Lee NeurOS and NeuroBlocks - a neural/cognitive operationg system and building blocks [Online] // Science Direct. - 2015. - http://www.sciencedirect.com/science/article/pii/S2212683X14000747.

Seung Sebastian Connectome [Book]. - New York, NY : Houghton Mifflin Harcourt, 2013. - First Mariner Books.