Biologically Inspired Cognitive Architectures (2014) **10**, 51–60

Available at www.sciencedirect.com
ScienceDirect
journal homepage: www.elsevier.com/locate/bica

Alleviating the curse of dimensionality — A psychologically-inspired approach

Vladislav D. Veksler [*], **Kevin A. Gluck, Christopher W. Myers, Jack Harris, Thomas Mielke**

Air Force Research Laboratory, Wright-Patterson AFB, USA

Received 7 November 2014; accepted 7 November 2014

KEYWORDS
Combinatorics;
Function approximation;
Chunking;
Unitization;
Configural-cue;
EPAM;
ACT-R;
Reinforcement learning;
Memory activation

Abstract
Various combinations of perceptual features are relevant for learning and action-selection. However, the storage of all possible feature combinations presents computationally impractical, and psychologically implausible, memory requirements in non-trivial environments due to a state-space explosion. Some psychological models suggest that feature combinations, or chunks, should be generated at a conservative rate (Feigenbaum and Simon, 1984). Other models suggest that chunk retrieval is based on statistical regularities in the environment, i.e. recency and frequency (Anderson and Schooler, 1991). We present a computational model for chunk learning based on these two principles, and demonstrate how combining these principles alleviates state-space explosion, producing exponential memory savings while maintaining a high level of performance.

Introduction

Decision making depends on the set of features perceived at decision time. Heart attack diagnoses depend on patient symptoms, such as chest pain and electrocardiogram readings; cyber attack detection and response depends on features of network activity; and the decision to break or turn in a critical driving situation depends on the sizes, locations, and direction vectors of nearby cars, pedestrians, and other obstacles. Whether the goal is to understand, predict, or aid human decision-making, or whether it is to achieve human-level performance in complex environments, inferring state-representation from perceived features is an important problem in Cognitive Science and Artificial Intelligence.

Cognitive architectures are often based on production systems (Anderson, 1993; Laird, 2012), where each production is a rule that specifies a condition (state) and an action to be fired whenever this condition is met. One of the greatest difficulties in the development of a cognitive model is in accounting for all the states that the model could encounter in a given task-environment. The inability to account for all

[*] Corresponding author.
E-mail address: vdv718@gmail.com (V.D. Veksler).

http://dx.doi.org/10.1016/j.bica.2014.11.007
2212-683X/© 2014 Elsevier B.V. All rights reserved.

potential states generates brittle models that halt in the face of error.

Autonomous agents in machine learning are often based on Reinforcement Learning (RL) (Sutton & Barto, 1998), taking a similar approach to production systems, but assuming all possible rules, or state-action pairs. That is, rather than specifying which rules are appropriate in a given environment, RL agents contain all possible state-action pairs in a lookup table, and select which actions to fire based on prior reward feedback recorded for each state-action pair. However, in environments where each state comprises numerous perceptual features, treating each unique combination of percepts as a state (lookup-based RL) is extremely inefficient. As Sutton and Barto (1998) point out, ''The problem is not just the memory needed for large tables, but the time and data needed to fill them accurately. In other words, the key issue is that of *generalization*.'' For example, a lookup-based RL agent may learn that eating red and green apples is rewarding, but will fail to generalize, and will produce random-level behavior when encountering a yellow apple.

Rather than treating each unique input as a state, it is possible to treat each perceptual feature as a state (feature-based RL). Assuming a world where perceptual features may be {red, green, yellow, brown, apple, chocolate, ...}, rather than reinforcing the action of eating for red apples and green apples, a feature-based RL agent would reinforce the red-eat, green-eat, and apple-eat state-action pairs. This would work well for the ability to generalize to yellow apples, having learned positive reinforcement for the apple-eat rule. However, such an agent could not learn about rule exceptions. For example, let us assume that the apple category has an exception, and brown apples in this world do not taste good (whereas other brown objects, such as chocolate, do taste good). Feature-based RL would fail to learn about the brown apple instance, or any such feature combination (e.g. the XOR problem).

Hand-coded models are brittle, lookup-based agents cannot generalize, and feature-based agents cannot learn exceptions. A brown apple is neither just brown, nor just an apple, nor just a brown apple. It is all of these things simultaneously, and any of these representations may be important for both learning and action-selection. Identifying the object as a brown-apple may be inefficient, and identifying it as just an apple may be misleading. Concurrent representation of all features combinations, a.k.a. chunks[1] or configural-cues (Gluck & Bower, 1988b; Wagner & Rescorla, 1972), would allow for learning of both generic rules (e.g. apples taste good) and exceptions to those rules (e.g. brown apples are spoiled).

Indeed, each set of perceptual features may potentially be recognized as all possible combinations of those features. Perceptual input $\{large, square, white\}$ may be represented as seven different states: $\{large\}, \{square\}, \{white\}, \{large, square\}, \{large, white\}, \{square, white\}$, and $\{large, square, white\}$. The problem with such representation is that too many memory chunks would be required in complex environments. A mere ten binary perceptual inputs (e.g. black vs white, large vs small) will require 59,048 chunks to be present in memory.[2] If each perceptual input allowed for five possible values (e.g. black, dark-gray, gray, light-gray, white), ten such input dimensions would result in almost ten million chunks. Twenty such perceptual dimensions would result in 95 trillion chunks. One hundred inputs with ten values per input would result in more chunks than there are atoms in the universe. The exponential growth of memory based on combinations of perceptual features is referred to as the state-space explosion problem, or the curse of dimensionality (Bellman, 1961).

To be clear, the problems with storing all possible chunk combinations, hand-coding models, or using lookup-based and feature-based agents are all well-known. These alternatives are presented here (1) to point out that ultimately we would like computational agents to learn generic situational rules, as well as exceptions to those situations, as well as exceptions to those exceptions, and so on, and (2) to highlight the difficulty with achieving this behavior. In practice, generalization for RL agents is done via one of many existing *function approximation* techniques.[3] These include neural networks, support vector machines, coarse coding (e.g. CMAC or tiling), decision trees, sparse distributed memory, radial basis function networks, and case-based reasoning (a.k.a instance-based or memory-based) methods (Kaelbling, Littman, & Moore, 1996; Sutton & Barto, 1998). Each of these methods provides advantages under specific conditions. Decision trees assume generic rules, and then gradually learn rule exceptions, but cannot learn in environments where feature-combinations, rather than features themselves, are predictive of performance (Kaelbling et al., 1996). Case-based reasoning methods (e.g. k-nearest neighbor) provide a way to account for all potential rules and exceptions, but their memory requirements approach those of lookup-based RL in persistent environments (Ratitch & Precup, 2004). Other methods greatly reduce memory demands, but require *a priori* knowledge about the task-environment (e.g. total number of rules needed to solve a task) (Kaelbling et al., 1996; Ratitch & Precup, 2004).

[1] We use the term chunk to refer to perceptual chunks, as is the case in EPAM/CHREST (Feigenbaum & Simon, 1984; Gobet et al., 2001). The term *chunk* has a slightly different use in the SOAR literature (Laird, 2012), referring to the creation of a production based on a resolved impasse — this is not the definition that we adopt in this paper.

[2] Given n features (e.g. $large, square, white$), we can create a chunk for every combination of feature presence and absence ($\{large\}, \{square\}, \{white\}, \{large, square\}, \{large, white\}, \{square, white\}$, and $\{large, square, white\}$). If we represent feature presence as a 1 and feature absence as a 0, we can represent each chunk as a binary number, and the total number of possible chunks is the total number of possible binary numbers, minus the blank chunk, which is $2^n - 1$. When each feature dimension can have two potential values, the total number of possible chunks is $3^n - 1$. With $k - 1$ possible values on n feature dimensions, we can have at most $k^n - 1$ possible chunks to represent all potential feature combinations.

[3] Function approximation has two purposes, dimensionality reduction and discretization of a continuous state-space. In this paper we assume a pre-discretized state-space. In the case of a continuos state-space, single-dimension function approximation (which is a much more tractable problem than multi-dimensional function approximation) may be done for each input dimension, and the conservative-rational mechanism proposed in this paper may be employed for dealing with the high dimensionality.

It is safe to say that the biological brain is able to cope with generic rules and exceptions to those rules across ranges of persistent tasks with varying requirements, all while avoiding a memory explosion. Thus, it may be prudent to borrow from psychological models for resolving this computational problem.

In this paper we propose the *conservative-rational* process for gradual chunk learning based on widely supported principles of human memory, and examine how the proposed mechanism helps to alleviate the curse of dimensionality in state-representation. Specifically, we propose that memory explosion is mitigated if new chunks are created via the union of existing memory chunks (1) only when none of the existing memory chunks fully represent the current state (similar to *familiarization* in EPAM/CHREST; Feigenbaum & Simon, 1984; Gobet et al., 2001), and (2) only when the activation of existing chunks goes above threshold, where activation is based on statistical regularities in the environment (i.e. recency and frequency; Anderson & Lebiere, 1998; Anderson & Schooler, 1991).

The rest of this paper presents psychological evidence for chunk-based memory representation and gradual chunk learning, introduces the proposed chunk-learning mechanism, and describes two simulations, demonstrating how the proposed mechanism alleviates state-space explosion. Simulation 1 presents evidence that the proposed mechanism aids in alleviating state-space explosion, especially in noisy environments. Simulation 2 examines performance-memory trade-off in a generic decision environment, presenting evidence that the proposed mechanism provides orders of magnitude in memory savings while maintaining a high level of performance.

Psychological validity

There is much evidence in the psychological literature that human memory contains representations of perceptual features, as well as various sets of these features as unique memory chunks (or their functional equivalent). Having a single memory chunk for representing a given combination of features allows for faster recognition of current state and faster action selection (Goldstone, 2000). Chunk representation also aids in the ability to store greater amounts of information in working memory (Anders Ericsson, Delaney, Weaver, & Mahadevan, 2004). A commonly cited example of this phenomenon involves improved recognition of displayed chess positions by chess experts compared to novices (Gobet, 1998; Chase & Simon, 1973). The theory is that experts will have been exposed to more chess situations, and will have learned many chunks for representing complex states on the chess board. Thus a chess expert will need fewer chunks to represent a chess position than a novice, who may have to memorize the position as a series of figures and their locations. This theory is further supported by the fact that experts are no better than novices in recall of chess boards where pieces are placed randomly (rather than legal positions).

Recognition of a set of perceptual features as multiple concurrently active chunks can account for a wide range of behavioral phenomena, including classical conditioning, base-rate neglect, and categorization (Gluck & Bower, 1988a, 1988b; Pearce, 1994). Veksler et al. (2007) integrated such representation with RL to account for category learning in the classic Shepard, Hovland, & Jenkins (SHJ) study (Shepard, Hovland, & Jenkins, 1961).

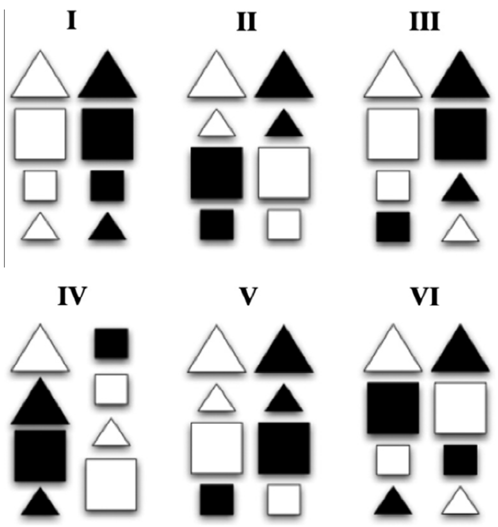

Fig. 1 An example of the six types of categorization problems from the Shepard, Hovland, & Jenkins task. For each type, subjects must learn that the items in the left column are one category, whereas the items in the right column are another.

In the SHJ paradigm participants are presented with one of eight objects varying across three binary dimensions (e.g. small/large, black/white, triangle/square), and have to learn which objects go into which of two categories. Given this setup, there are only six different types of possible category breakdowns (see Fig. 1). When feedback is provided as to whether the participant categorization response is correct, human performance indicates that category structure I was easiest to learn, followed by II, then III/IV/V, with problem type VI being the most difficult (Love, 2002; Nosofsky, Gluck, Palmeri, McKinley, & Glauthier, 1994; Shepard et al., 1961; Smith, Minda, & Washburn, 2004). Note that category structure VI requires lookup-based representation for any performance gains, as none of the perceptual features have response-predictive validity. For category structure I, on the other hand, a single perceptual feature may be used to determine the correct category response, and feature-based representation would be useful for efficient performance on this problem. Category structures II through V require all possible chunk representations, so that generic rules, as well as exceptions to those rules may be learned. The Veksler, Gray, & Schoelles model included state-action pairs for all possible chunks, and could thus account for human performance on this task, displaying the ability to generalize, and to learn exceptions where appropriate. In a variation on the SHJ paradigm humans can learn the six category structures through observation, rather than through trial-and-error (Love, 2002). Given this setup, results indicate that category structure II is more difficult to learn than IV (Love did not test category structures III and V in this study). The model accounted for this result when the exploration parameter was turned down (essentially turning off the trial-and-error learning in favor of observation-based learning).

To be clear, Veksler et al. (2007) did not include chunk learning in their model, instead assuming that all possible chunks in that task existed in human memory (which may be a reasonable assumption given the simple stimuli in SHJ). However, there is also much psychological evidence that humans do not hold memory representations for all potential combinations of situational features. Rather, features get merged over time to make more and more complex object representations. There is evidence of this phenomenon in the context of perceptual unitization within hour-long psychological experiments (Goldstone, 2000), as well as in life-long acquisition of subject-matter expertise (Feigenbaum & Simon, 1984).

The next section describes a gradual chunk learning mechanism inspired by two psychological models of learning — EPAM's familiarization (Feigenbaum & Simon, 1984) and ACT-R's base-level learning (Anderson & Lebiere, 1998; Anderson & Schooler, 1991). The simulation sections describe how these psychologically-inspired principles help to alleviate the state-space explosion and, integrated with RL, provide performance advantages over lookup-based and feature-based agents.

Gradual chunk learning via the conservative-rational principles of memory

According to Simon's early computational models of human learning (EPAM; Feigenbaum & Simon, 1984), agent state may be represented as a chunk — a mathematical set of perceptual features (e.g. $\{large, white, square\}$), and larger, more specific chunks are generated by adding more features to existing chunks (i.e. *familiarization* or *chunking*; $\{large, white, square\} \cup \{rotated\} = \{large, white, square, rotated\}$). Familiarization in EPAM happens in a *conservative* manner, gradually, and only when the model does not yet contain a chunk that includes the entire set of perceptual features in the observed state. That is, early on states are represented in memory as individual features, then as small chunks, and then, gradually, as chunks of greater and greater specificity.[4]

Anderson's work on memory, in turn, suggests that chunks can be retrieved when their activation is above threshold. Chunk activation is based on the statistical properties of the environment (*rational* analysis; Anderson & Lebiere, 1998; Anderson & Schooler, 1991). If the set of features that make up a chunk co-occur in recent history, and/or co-occur frequently, this chunk is deemed relevant, and thus more likely to be retrieved.

Combining the *conservative* chunk growth and the *rational* memory retrieval principles together, we propose a conservative-rational chunk learning model. In general, the model may be described as such: (1) new chunks are generated gradually via the union of retrieved existing memory chunks only when none of the existing chunks fully represent the current state, (2) chunks are retrievable only when their activation level goes above threshold, (3) the more active chunks are more likely to be retrieved, and (4) chunk activation level is based on the recency and frequency of chunk occurrence in the environment (or, rather, the occurrence of the features that make up that particular chunk).

More formally, when the conservative-rational model is initiated it contains one base-level chunk for every perceptual feature that may be observed in the environment, such that if there are x possible features, there are also x unique chunks in memory, M, each containing its respective perceptual feature, $M = \{\{feature_1\}\ldots\{feature_x\}\}$. When the model is exposed to some perceptual input containing a set of features, F, activation is added to each existing memory chunk, s, in set:

$$S = \{s | s \in M, s \subseteq F\} \tag{1}$$

Chunk activation decays over time. At any point the activation of chunk i, A_i, can be calculated as follows (vis-a-vis base-level activation formula from ACT-R; Anderson & Lebiere, 1998):

$$A_i = \sum_{j=1}^{n} t_j^{-d} + \sigma \tag{2}$$

where n is the number of presentations of chunk i, t_j is the time elapsed since the jth presentation of i (in steps; $t_j \geqslant 1$)[5], d is the decay rate,[6] and σ is noise (a small random number added to chunk activation to add stochasticity to the model).

At each step, two chunks with the highest activations, a and b, are selected from the set of chunks in S where no chunk is a subset of any other chunk in $S, \{i \in S | \forall_{j \in S} i \neg \subset j\}$ (e.g. if the set of chunks in S is {{white}, {large}, {rotated}, {oval}, {white, large}, {white, large, rotated}}, then {white, large, rotated} and {oval} will be the only candidates for creating a new chunk). If the activations of a and b are both above the retrieval threshold parameter, ρ, a new chunk is added to memory via the union of a and $b, M = M \cup \{a \cup b\}$.

Note that for all chunks $s \in S$ the most recent elapsed time t_j is 1, and $A_s >= 1$. Thus, when $\rho <= 1$ this model is just conservative and not rational, since every chunk in S is above threshold, regardless of recency and frequency of prior activations.

Simulation 1: alleviating state-space explosion

This simulation examines the degree to which the conservative-rational chunk learning principles alleviate state-space explosion. Via the *conservative* principle, the proposed

[4] EPAM/CHREST models actually replace smaller chunks with larger ones during the familiarization process. An additional learning process, discrimination, is used to recreate smaller chunks if these are deemed needed later. For current purposes we borrow only the EPAM chunk generation principles, and not the processes of chunk deletion and recreation, as these are not compatible with the Reinforcement Learning integration employed in Simulation 2.

[5] This model implementation is event-based, counting each perception-action cycle as a discrete step rather than in continuous time. Assuming constant cycle times, these are computationally equivalent. Elapsed time begins at 1, because at $t_j = 0, t_j^{-d} = \infty$.

[6] For simplicity, in simulations below decay rate is set to 1.0, which is equivalent to hyperbolic decay. Different decay rates would not qualitatively alter simulation results. Decay rate, d, is inversely related to the retrieval threshold parameter, ρ, such that if d was lower, similar results may be achieved with a higher ρ value.

Table 1 Average number of chunks created after 100,000 steps in static, constrained, and chaotic 5-dimensional and 10-dimensional environments.

	5 input dimensions			10 input dimensions		
	Static	Constrained	Chaotic	Static	Constrained	Chaotic
All-chunks	31	9928	32,767	1023	1,528,477	1,073,741,823
$\rho = 1.0$	9	4988	26,895	19	19,183	85,841
$\rho = 1.5$	9	2430	856	19	8947	2736
$\rho = 2.0$	9	1238	228	19	4293	380
$\rho = 2.5$	9	652	35	19	1961	70

model should only create chunks when none of its current memory chunks match the presented stimulus. In this manner, the model should create less than all-possible chunks unless it is exposed to all potential feature combinations multiple times. Via the *rational* principle, the model should only create chunks based on stimuli that appear frequently and continuously, ignoring accidental feature co-occurrences. In this manner, the model should be able to separate signal from noise, creating fewer chunks in non-sensical environments than in ecologically-constrained ones.

We ran the proposed chunk-learning model for 100,000 steps in environments with five and ten input dimensions, varying the retrieval threshold parameter, ρ from 1.0 (no recency/frequency effect) to 2.5. For both five- and ten-dimensional inputs we examined chunk growth for static, constrained, and chaotic environments (see environment descriptions below). The results, displayed as averages from ten model runs and compared to a model that created all possible chunks, are displayed in Table 1 and Fig. 2 (y-axis in log-scale).

The static environment simulation consisted of repeatedly exposing the model to a set of five features in a five-dimensional environment (e.g. step 1: $\{a, b, c, d, e\}$, step 2: $\{a, b, c, d, e\}$, …step 100,000: $\{a, b, c, d, e\}$), and ten features in a ten-dimensional one. In these environments there are 31 possible chunks to be created with 5 input dimensions, and 1023 possible chunks to be created with 10 input dimensions. However, the proposed model (at all ρ values) creates only 9 and 19 chunks in the 5- and 10- dimensional environments, respectively.

This happens because chunks in the conservative-rational model are created only when necessary. Thus, in the 5-dimensional static environment where the model is constantly presented the input $\{a, b, c, d, e\}$, the model will be initiated with five chunks — $\{a\}, \{b\}, \{c\}, \{d\}, \{e\}$ — and will create only four more chunks (for example, it might create chunks in the following sequence $\{b, c\}, \{b, c, e\}$, $\{a, b, c, e\}$, and $\{a, b, c, d, e\}$). No other chunks will be generated because the chunk $\{a, b, c, d, e\}$ is sufficient to represent the state $\{a, b, c, d, e\}$.

In the chaotic environment each input dimension could take on one of seven unique values, or could be absent from the input state (with a minimum of two input features present at each step). At each step the number of features in the state was chosen at random, with a minimum value of 2 and a maximum value of 5 and 10 in the 5-dimensional and 10-dimensional environments, respectively. The feature value for each respective input dimension was a number between 1 and 7, chosen at random, with replacement. Thus, any feature combinations of length between 2 and 7 had the potential to be presented in this environment (e.g. in the 5-dimensional environment inputs could be $\{a:1, b:1, c:2\}, \{a:7, b:1\}, \{b:4, c:5, d:7, e:7\}$).

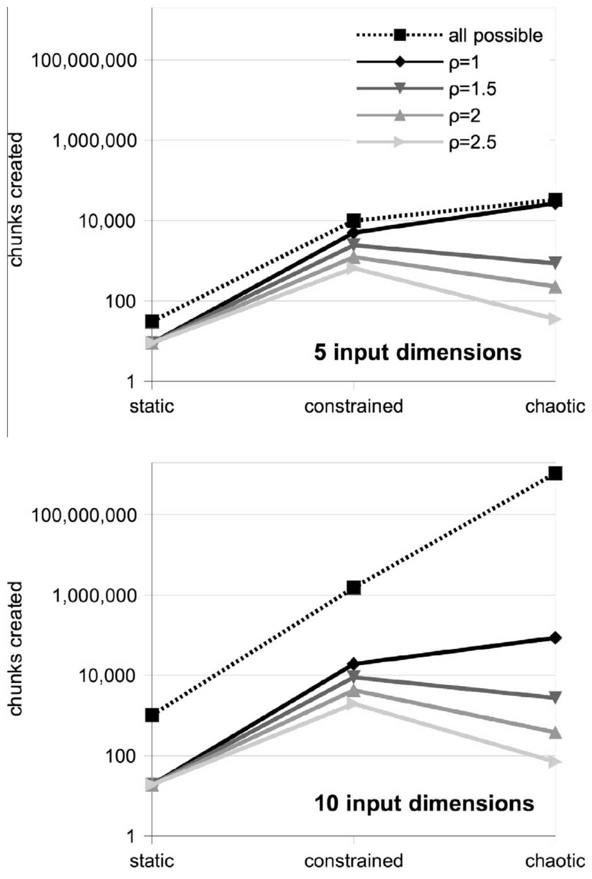

Fig. 2 Average number of chunks created after 100,000 steps in static, constrained, and chaotic 5-dimensional and 10-dimensional environments. Standard error is negligible.

Note that in the 10-dimensional chaotic environment the $\rho = 1$ model generates chunks at a pace of almost one per step. When the retrieval threshold is above 1.0, chunks are generated at a much more conservative pace. At $\rho = 2.5$ the model creates *no chunks* beyond the initial 35 and 70 base-level chunks for the 5-dimensional and 10-dimensional chaotic environments, respectively.

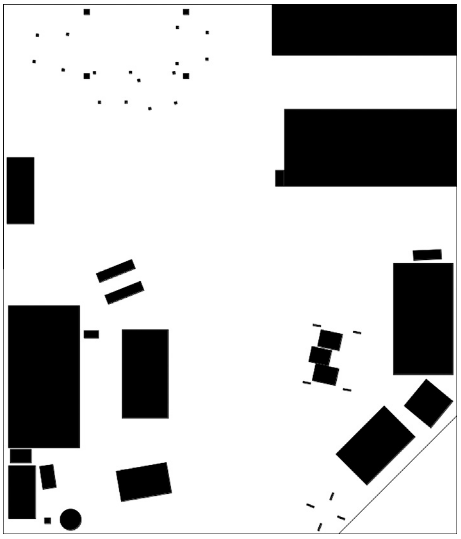

Fig. 3 Simulated room environment used in Simulation 1 (drawn to scale from an actual room). All filled areas represent obstacles at the robot sensor height (e.g. the twenty dots at the top-left of the figure are chair and table legs).

Finally, the constrained environment was a simulated random walk in a room with various furniture by a robot with 5 proximity sensors (at $0°, 10°, -10°, 45°, -45°$) or 10 proximity sensors (at $2°, -2°, 10°, -10°, 30°, -30°, 45°, -45°$).[7] Each proximity sensor reading was converted to an integer between 1 and 7. In this way the room walk had the potential to have similar complexity to the 5x7 and 10x7 chaotic environments, except that it was constrained by actual simulated robot movements.[8] The room layout is displayed in Fig. 3.

Note that as the environment becomes more complex (comprising more unique input vectors), from static to constrained to chaotic, the number of chunks created in the all-chunks and $\rho = 1$ models increases. However, the $\rho > 1$ (recency/frequency) models create *fewer* chunks in the chaotic than in the constrained environments. In other words, rational memory principles help to derive signal from noise, and allocate memory only to the most relevant feature combinations. Intuitively this seems like correct behavior, and we expect that humans will learn less in an environment that is completely nonsensical than in one that has some regularity and continuity.

This type of ''rational'' chunk generation is much different than merely creating chunks at a conservative rate. Even if chunks are only created once every few seconds, or only a small proportion of the time, chunk growth would be a linear function of time, creating as many chunks from incidental feature co-occurrences in the chaotic environment as from the co-occurring features in the more ecologically-constrained environment. Alternatively, the recency/frequency principles provide an informed approach to chunk learning, based on the likelihood of chunk occurrence. Thus, the proposed model creates *fewer* chunks in noisy environments where the potential for state-space explosion is greatest.

More formally, with $k - 1$ possible values on n perceptual feature dimensions, space complexity for a model that creates all possible chunks is $O(k^n)$. Space complexity for a lookup-based agent would also be $O(k^n)$ (in environments where each sensory input consists of exactly n perceptual features, lookup-based space complexity is $O(j^n)$, where $j = k - 1$). For case-based reasoning (or instance-based, memory-based) agents, space complexity grows linearly toward that of lookup-based agents. For the proposed model, when $\rho = 1$ (no recency/frequency), as time increases complexity grows toward $O(k^n)$, as well. However, when $\rho > 1$, space complexity is inversely proportional to the noise in the environment.

The probability of creating a new chunk in the conservative-rational model, P_{new}, by combining existing memory chunks a and b is $P(A_a > \rho) \times P(A_b > \rho)$, where A_a and A_b are activations for chunks a and b, respectively (calculated as shown in Eq. (2)). As ρ increases, as the probability of any two features co-occurring in the environment decreases, and as the lengths of chunks a and b increase, $P_{new} \to 0.0$, and the space complexity of the model approaches $O(nk)$. This is the reason why random feature co-occurrences are less likely to be learned than meaningful ones, and why the model creates zero new chunks at $\rho = 2.5$ in the nonsensical environment in this simulation.

Thus, if we were to add n noisy input sensors to the simulated 10-sensor robot described above, essentially combining the 10-dimensional constrained environment with an n-dimensional chaotic environment, we would not expect to see much memory growth between an environment with lower dimensionality, $n = 2$, and one with higher dimensionality, $n = 20$, or even $n = 200$. Indeed, running such a simulation reveals that space complexity for the conservative-rational model does not increase greatly with more noisy dimensions, nor does it increase greatly over time (see Fig. 4).

Simulation 2: performance-memory trade-off

The above simulation demonstrates how the conservative-rational principles of memory help to mitigate the state-space explosion in chunk learning, producing as much as 99.99% memory savings. An important question, however, is whether such a low number of chunks can produce a level of performance similar to a model that generates all possible chunks. To examine this, we integrated chunk-based memory with RL, and examined model performances in sample decision environments.

The integration of chunk-based memory with RL was implemented as follows. Given the set of active chunks, S (as defined in Eq. (1)), the action with the highest utility is executed, where we define action utility for each action, a, as

$$U_a = \sum_{s \in S} U_{sa} + N,$$

[7] All models in the 5-sensor environment were exposed to the same random walk, and all models in the 10-sensor environment were exposed to the same random walk.

[8] The five- and ten- septenary proximity sensor robot simulations were developed as a part of a parallel research effort. The five- and ten- dimensional static and chaotic environments were developed as analogs to the robot simulations for direct contrast in examining memory savings in the conservative-rational model.

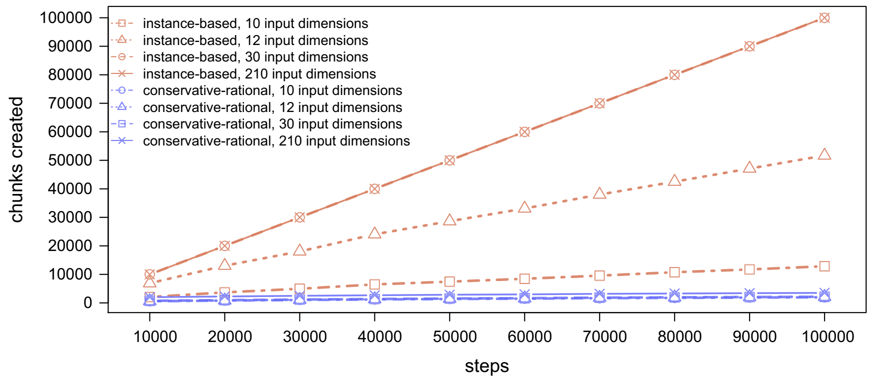

Fig. 4 Number of chunks created over 100,000 steps in the 10-dimensional constrained environment with n added noisy input dimensions, $n \in \{0, 2, 20, 200\}$, for case-based reasoning (instance-based) and conservative-rational ($\rho = 2.5$) models. In sufficiently complex environments (e.g. $n = 20, n = 200$) instance-based approaches generate a new chunk at every time-step, whereas the $\rho = 2.5$ memory growth is relatively flat. Results displayed as averages over ten model runs. Standard error is negligible.

where U_{sa} is the cached utility of state-action pair sa and N is small gaussian noise added for stochasticity. We used the Reinforcement Learning equation from ACT-R to update state-action pair utilities upon reward feedback, where the utility of each state-action pair is increased at time n by

$$\Delta U_{sa}(n) = \alpha[r(n) - U_{sa}(n-1)],$$

where $r(n)$ is the reward received at time n, and α is the learning rate parameter[9]. Because this is a single-step stimulus-response task, variant RL equations would not produce qualitatively different simulation results.

The environment generator we used for this simulation was developed to support research on binary decision environments (e.g. to send a patient to CCU or not; to classify network activity as a cyber-attack or not). The generator creates decision cases based on different binary feature distributions and outcome rules. We used the generator to create an environment approximating a generic threat detection task. On each trial in this environment each model had to classify a set of 10 or 12 binary perceptual features as either a threat or not a threat. Each model had to operate and adapt through four consecutive threat scenarios, each scenario lasting 500 trials. In the first scenario the threat could be identified via a combination of three randomly chosen perceptual features (e.g. when patient has red spots, and coughing, and reporting chest pain, classify as a threat). In the second, third, and fourth scenarios the threat could be identified via combinations of four, three, and four randomly chosen perceptual features, respectively. The probability of a threat on any given trial was 50%.[10]

The reward values for correctly classifying a threat and missing a threat were relatively high (+10 and −10, respectively), the reward value for correctly rejecting a given scenario as a valid threat was relatively low (+1), the value of sounding a false alarm was somewhere in between those two (−5).10 The models were evaluated based on the average reward per trial that they were able to achieve during the simulation.

Fig. 5 displays performance and number of chunks created in this simulation for five models: (1) a model with all possible chunks, (2) the standard lookup-based Reinforcement Learning model, where each distinct combination of input values was treated as a unique state, (3) a feature-based model, where no new chunks were created beyond the initial 20 chunks in the 10-dimensional and 24 chunks in the 12-dimensional decision environments (one chunk for each potential input feature), (4) a chunk learning model that employs no recency/frequency principles ($\rho = 1$), and (5) the full conservative-rational model ($\rho = 1.5$).

In the 10-dimensional environment the performance rank of the five models was as follows: the all-chunks model performed best (average reward of 5.2 per trial over 2000 trials), followed by the $\rho = 1$ model (−10.6%), then $\rho = 1.5$ (−13.2%), feature-based (−28.6%), and lookup-based models (−52.7%). In the 12-dimensional environment the performance rank of the five models was as follows: the all-chunks model performed best (average reward of 5.0 per trials over 2000 trials), followed by the $\rho = 1.5$ model (−13.3%), then $\rho = 1$ (−13.8%), feature-based (−31.4%), and lookup-based models (−91.1%). All models, except the all-chunks model, performed better in the second half of the simulation, as time was needed for training and chunk generation (see Table 2).

Fig. 6 displays the relationship between performance and memory consumption in this simulation for both 10- and 12-dimensional environments (y-axis in log-scale). Although it may be obvious that performance can be gained at the expense of memory, what may be less obvious is that (1) small performance gains require exponential memory growth, and (2) as the environment becomes more complex (from 10 dimensions to 12 dimensions) performance differences become less distinguishable while memory expenses become more distinguishable.

On a final note, the 10- and 12-dimensional binary environments are relatively simple. In more complex environments the all-chunks model is computationally infeasible due to state-space explosion. Additionally, the $\rho = 1$ and lookup-based models produce nearly one chunk per

[9] $\alpha = .1$ for this simulation. Different parameter values do not qualitatively alter reported simulation results.

[10] Simulation parameters (e.g. threat probabilities, reward values) are reported here for purposes of replicability. Main result trends do not vary greatly with different simulation parameters.

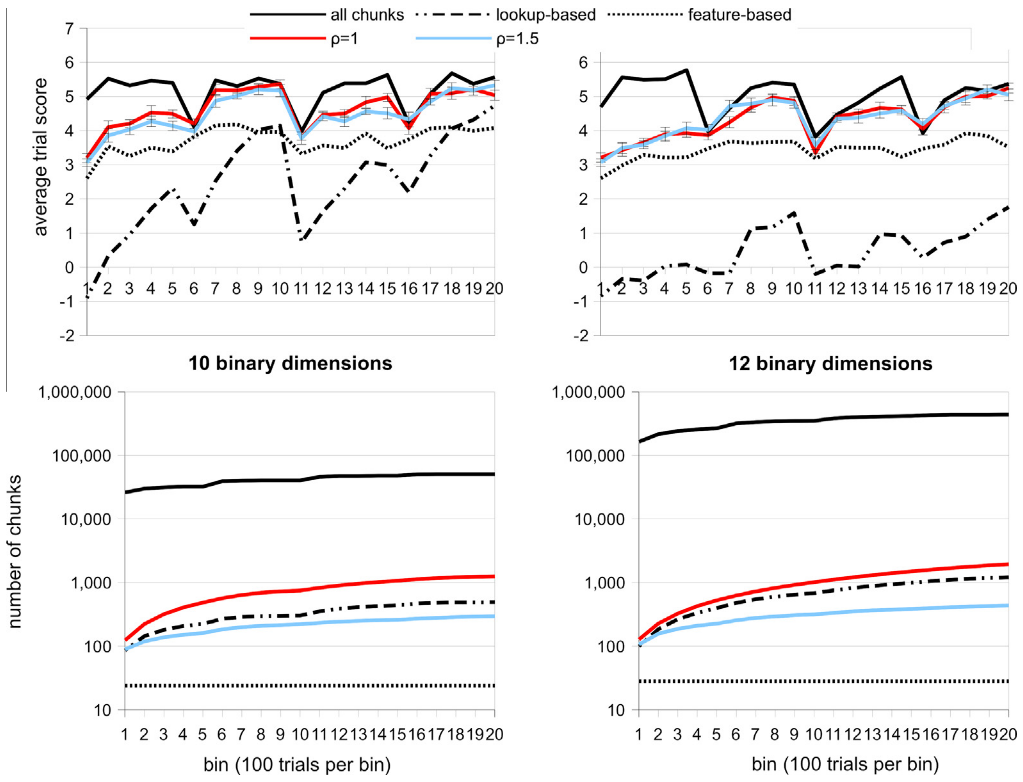

Fig. 5 Performance and memory growth over time in 10- and 12- binary dimension decision environments. Error bars represent standard error.

Table 2 Average performance loss in the second half of Simulation 2, relative to the all-chunks model.

	10-Dimensional (%)	12-Dimensional (%)
$\rho = 1.0$	−8.4	−6.1
$\rho = 1.5$	−9.6	−6.2
$\rho = 2.0$	−10.9	−9.3
$\rho = 2.5$	−15.6	−19.6
Feature-based	−26.5	−27.3
Lookup-based	−43.0	−86.0

time-step, which makes them computationally infeasible in more persistent high-dimensional environments. The more feasible feature-based model performance may be interpreted as ''good-enough'' given that it only requires one chunk per perceptual feature. Indeed, if memory limits were of extreme concern and perceptual features provided ''good-enough'' response validity, and if psychological validity was not the intended goal, a feature-based computational agent may be preferred. However, it should be highlighted that the conservative-rational model (1) bares more psychological validity, (2) performs at a very high level, showing only about 10% drop-off from the infeasible all-chunks model performance after training in the 10-dimensional environment, and about 6% drop-off in the 12-dimensional environment (feature-based loss is about 27% in both environments), and (3) generates less than 1,000 chunks to achieve this performance. To put this in perspective, the storage required in RL for 1000 states x 2 actions can be less than 8 KB. Given that even simple boards

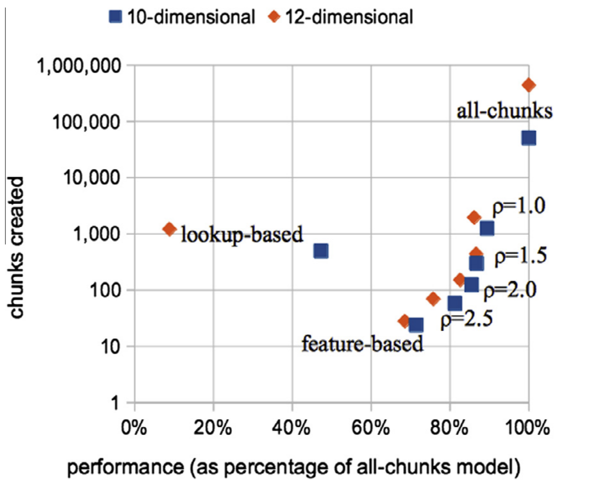

Fig. 6 Chunk number versus performance in 10- and 12-dimension binary decision environments.

currently offer higher memory limits (Arduino Due offers 96 KB of SRAM) it may be unnecessary and wasteful to opt for feature-based state representation over the conservative-rational chunk learning method.

Summary and discussion

Whether the goal is to understand, predict, or aid human decision-making, or whether it is to achieve human-level performance in complex environments, a computational model must be able to correctly interpret its state from perceived features. Hand-coding potential world states creates brittle models, lookup-based agents cannot generalize, and feature-based agents cannot learn exceptions.

There is much psychological evidence that states are represented as multiple varying configurations of perceptual features (Pearce, 1994). Configural (or chunk) representation provides many cognitive advantages, such as extending the size of working memory (Anders Ericsson et al., 2004), and providing representation for both generic rules and instance exceptions (Gluck & Bower, 1988b; Veksler et al., 2007). However, it is computationally implausible to hold all possible feature configurations in memory for environments of even modest complexity, because the number of chunks grows exponentially as a function of the number of perceptual features. This is termed the state-space explosion problem.

The state-space explosion problem is often addressed in machine learning literature via dimension-reduction techniques such as neural networks, decision trees, sparse distributed memory, and radial basis function networks (Kaelbling et al., 1996; Sutton & Barto, 1998). These techniques, however, either fail to capture configural state representations, or require varying amounts of *a priori* information about the problem space (Kaelbling et al., 1996; Ratitch & Precup, 2004). More flexible methods, such as case-based reasoning, fail to address memory explosion in persistent environments (Ratitch & Precup, 2004).

We propose a *conservative-rational* mechanism for alleviating state-space explosion based on two psychological models, EPAM familiarization (Feigenbaum & Simon, 1984), and ACT-R base-level learning (Anderson & Lebiere, 1998). Simulation 1 demonstrates how the proposed model alleviates the state-space explosion, especially in noisy environments. Specifically, when chunk generation is done gradually, and only when necessary (vis-a-vis EPAM familiarization), less chunks are generated across all environments. Moreover, when chunk generation is based on statistical regularities in the environment (vis-a-vis recency/frequency principles in ACT-R base-level learning), it aids in distinguishing signal from noise, sorting meaningful co-occurrences from accidental ones.

Simulation 2 examines performance-memory tradeoffs in generic binary decision environments. Results indicate that small improvements in performance require exponential sacrifices in memory use. Compared with a model that created all possible chunks, the conservative-rational model maintained a high level of performance, while providing orders of magnitude in memory savings. It is worth noting that the all-chunks model is computationally infeasible in more complex environments, and that the conservative-rational model provides far better performance than the computationally feasible lookup-based and feature-based models.

State-representation, state-space explosion, and memory-size in general are among chief concerns for Artificial Intelligence and Cognitive Architectures (Bolch, Greiner, de Meer, & Trivedi, 2006; Douglass et al., 2009; Sutton & Barto, 1998; Uther & Veloso, 1998). The current work focuses on limiting memory expansion based on recency/frequency principles, expanding only the memory elements with the highest activation. An alternative approach, proposed by Derbinsky and Laird (2013), employs the recency/frequency principles to delete memory elements with low activation. Future work will focus on employing these principles in concert, to further decrease memory size.

The number of connections needed between chunks, actions, and rewards may be another source of severe memory growth. The time required at each cycle for learning and action-selection processes would increase in proportion with the number of connections in memory. In RL, where the focus is on learning State-Action-Reward transitions and employing these in action-selection, the number of stored connections would be a linear function of the number of chunks. For systems that focus on State-Action-State transitions (Veksler, Gray, & Schoelles, 2013; Voicu & Schmajuk, 2002), or systems that focus on both transition types (Gläscher, Daw, Dayan, & O'Doherty, 2010; Veksler, Myers, & Gluck, 2014), it would be an exponential function. In addition to minimizing the number of chunks in memory, future work will explore how the principles discussed in this paper may be employed to create connections and to prune them in a more conservative and rational manner.

Acknowledgements

This research was funded by Grant #13RH06COR from the Air Force Office of Scientific Research, and partly conducted while the first author held a National Research Council Research Associateship with the Cognitive Models and Agents branch at the Air Force Research Laboratory.

References

Anders Ericsson, K., Delaney, Peter F., Weaver, George, & Mahadevan, Rajan (2004). Uncovering the structure of a memorist's superior basic memory capacity. *Cognitive Psychology, 49*(3), 191—237.

Anderson, John R. (1993). *Rules of the mind*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Anderson, John R., & Lebiere, Christian (1998). *The atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum Associates Publishers.

Anderson, John R., & Schooler, L. J. (1991). Reflections of the environment in memory. *Psychological Science*, 2(6), 396—408.

Bellman, Richard E. (1961). *Adaptive control processes: A guided tour*. Princeton, NJ: Princeton University Press.

Bolch, Gunter, Greiner, Stefan, de Meer, Hermann, & Trivedi, Kishor Shridharbhai (2006). *Queueing networks and Markov chains: Modeling and performance evaluation with computer science applications*. John Wiley & Sons.

Chase, William G., & Simon, Herbert A. (1973). Perception in chess. *Cognitive Psychology, 4*(1), 55—81.

Derbinsky, Nate, & Laird, J. E. (2013). Effective and efficient forgetting of learned knowledge in soar's working and procedural memories. *Cognitive Systems Research, 24*, 104—113.

Scott Douglass, Jerry Ball, and Stuart Rodgers. (2009). Large declarative memories in ACT-R. In *9th International conference of cognitive modeling, ICCM 2009. DTIC document*.

Feigenbaum, E. A., & Simon, H. A. (1984). EPAM-like models of recognition and learning. *Cognitive Science, 8*(4), 305—336.

Gläscher, J., Daw, N., Dayan, P., & O'Doherty, J. P. (2010). States versus rewards: Dissociable neural prediction error signals underlying model-based and model-free reinforcement learning. *Neuron, 66*(4), 585—595.

Gluck, M. A., & Bower, G. H. (1988a). Evaluating an adaptive network model of human learning. *Journal of Memory and Language, 27*(2), 166—195.

Gluck, M. A., & Bower, G. H. (1988b). From conditioning to category learning — an adaptive network model. *Journal of Experimental Psychology-General, 117*(3), 227—247.

Gobet, F. (1998). Expert memory: S comparison of four theories. *Cognition, 66*(2), 115—152.

Gobet, Fernand, Lane, Peter C. R., Croker, Steve, Cheng, Peter C. H., Jones, Gary, Oliver, Iain, et al (2001). Chunking mechanisms in human learning. *Trends in Cognitive Sciences, 5*(6), 236—243.

Goldstone, Robert L. (2000). Unitization during category learning. *Journal of Experimental Psychology: Human Perception and Performance, 26*(1), 86.

Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research, 4*, 237—285.

Laird, J. E. (2012). *The Soar cognitive architecture*. MIT Press.

Love, B. C. (2002). Comparing supervised and unsupervised category learning. *Psychonomic Bulletin and Review, 9*(4), 829—835.

Nosofsky, R. M., Gluck, M. A., Palmeri, T. J., McKinley, S. C., & Glauthier, P. (1994). Comparing models of rule-based classification learning: A replication and extension of Shepard, Hovland, and Jenkins (1961). *Memory and Cognition, 22*(3), 352—369.

Pearce, J. M. (1994). Similarity and discrimination: A selective review and a connectionist model. *Psychol Review, 101*(4), 587—607.

Bohdana Ratitch and Doina Precup. (2004). Sparse distributed memories for on-line value-based reinforcement learning. In *Machine learning: ECML 2004*, pp. 347—358.

Shepard, Roger N., Hovland, C. I., & Jenkins, H. M. (1961). Learning and memorization of classifications. *Psychological Monographs: General and Applied, 75*, 1—42.

Smith, J. D., Minda, J. P., & Washburn, D. A. (2004). Category learning in rhesus monkeys: a study of the Shepard, Hovland, and Jenkins (1961) tasks. *The Journal of Experimental Psychology: General, 133*(3), 398—414.

Sutton, Richard S., & Barto, Andrew G. (1998). *Reinforcement learning: an introduction*. Cambridge, Massachusetts: The MIT Press.

William T. B. Uther and Manuela M. Veloso. (1998). Tree based discretization for continuous state space reinforcement learning. In *AAAI/IAAI*, pp. 769—774.

Vladislav D .Veksler, W. D. Gray, and M. J. Schoelles. (2007). Categorization and reinforcement learning: state identification in reinforcement learning and network reinforcement learning. In *29th annual meeting of the cognitive science society, CogSci07, Nashville, TN*.

Veksler, Vladislav D., Gray, W. D., & Schoelles, M. J. (2013). Goal-proximity decision making. *Cognitive Science, 37*(4), 757—774.

Veksler, Vladislav D., Myers, C. W., & Gluck, K. A. (2014). SAwSu: An integrated model of associative and reinforcement learning. *Cognitive Science, 38*(3), 580—598.

Voicu, H., & Schmajuk, N. (2002). Latent learning, shortcuts and detours: a computational model. *Behavioural Processes, 59*(2), 67—86.

Wagner, A. R., & Rescorla, R. A. (1972). Inhibition in Pavlovian conditioning: Application of a theory. *Inhibition and Learning*, 301—336.