Using Chunks to Categorise Chess Positions

Peter C.R. Lane and Fernand Gobet

Abstract Expert computer performances in domains such as chess are achieved using techniques different from those which can be used by a human expert. The match between Gary Kasparov and Deep Blue shows that human expertise is able to balance an eight-magnitude difference in computational speed. Theories of human expertise, in particular the chunking and template theories, provide detailed computational models of human long-term memory, how it is acquired and retrieved. We extend an implementation of the template theory, CHREST, to support the learning and retrieval of categorisations of chess positions. Our extended model provides equivalent performance to a support-vector machine in categorising chess positions into opening, and reveals how learning for retrieval relates to learning for content.

1 Introduction

Building a machine which exhibits intelligent behaviour has been a long sought-after dream, particularly in the fields of artificial intelligence (AI) and cognitive science. Two main approaches may be identified. The first is to use whatever techniques are offered by computer science and AI, including brute force, to create artifacts that behave in an intelligent way. The second is to develop computational architectures that closely simulate human behaviour in a variety of domains. The difference between the two approaches separates the study of AI and cognitive science:

> AI can have two purposes. One is to use the power of computers to augment human thinking, just as we use motors to augment human or horse power. Robotics and expert systems are

Peter C.R. Lane
School of Computer Science, University of Hertfordshire, College Lane, Hatfield AL10 9AB, UK
e-mail: peter.lane@bcs.org.uk

Fernand Gobet
School of Social Sciences, Brunel University, Uxbridge UX3 8BH, UK
e-mail: fernand.gobet@brunel.ac.uk

> major branches of that. The other is to use a computer's artificial intelligence to understand how humans think. In a humanoid way. If you test your programs not merely by what they can accomplish, but how they accomplish it, then you're really doing cognitive science; you're using AI to understand the human mind. (Herbert Simon, in an interview with Doug Stewart [37].)

In specific domains, AI can produce incredible performance. A long-standing goal of mathematicians and AI researchers was to develop a mechanical program that could play chess: the basic pattern for search algorithms was determined in the 1950s [36, 38], and recently algorithms such as Monte Carlo Tree Search [7] have dominated game-playing research. Humans, of course, are more selective in their search for good moves, and interactions of the search tree with other forms of knowledge were explored by Berliner [2] and Michie [31], amongst others. A more complete overview of this history can be found in Chapter 2 of [19].

In 1996 and 1997 IBM arranged two matches between its Deep Blue computer and the then World Chess Champion Gary Kasparov; Kasparov won the first match and Deep Blue won the second match – effectively a tie overall. These matches show the gulf between what we know of human intelligence and AI. The rapid indexing and sifting of an extensive pool of prior knowledge by the human counter-balanced an eight-magnitude difference in computational ability: where the human might calculate around 100 moves in 3 minutes, the computer would calculate of the order of 100 million moves per second. Chess has been described as the 'drosophilia of psychology' [6], and in these matches we see the need and opportunity for theories of intelligence to understand how humans efficiently perceive a stimulus, such as a chess board, and retrieve information suitable for problem solving.

We present a cognitive-science approach to solving this problem. Cognitive science has a long tradition of attempting to understand intelligence using an approach based around simulation (some of this history is discussed in [1, 16]). The earliest systematic work in this area is perhaps that of Newell and Simon [33]. The use of computational models for understanding cognition has grown over time, as their use offers many advantages. For example, their implementation as computer programs ensures a high degree of precision, and offers a sufficiency proof that the mechanisms proposed can carry out the tasks under study – something obviously desirable if practical artificial intelligence is the goal. The extent to which success is reached in simulating actual human behaviour can be assessed by using measures such as eye movements, reaction times, and error patterns, as well as the protocol analyses used by the earlier researchers, and, in more recent times, MRI scans of the working brain. Recent publications have emphasised the popularity of this approach, and recent summaries are available in [29, 35].

As a concrete task, we consider the game of chess, and how chess players remember and retrieve information about specific chess positions. Chess is an ideal domain for studying complex learning because it provides a well-defined set of stimuli (the different board positions), there are numerous examples of games readily available for study, and, in particular, chess players are ranked on a numeric scale allowing gradations in levels of expertise to be identified in a quantitative manner. In competitive chess, players are limited in their thinking time, and success depends on an

efficient retrieval of information from their prior study and experience to help understand the position in front of them. Chess positions present various problems to the player, enabling the researcher to probe the varying contributions of calculation and intuition in the problem-solving process.

We examine how a cognitive architecture based on the influential chunking and template theories [6, 20, 22] constructs and uses representations for visual stimuli using an incremental-learning technique. The form of the representation is governed both by physical demands of the domain, such as locality, and also by procedural requirements for an efficient indexing mechanism and to interact with visual attention. CHREST (Chunk Hierarchy and REtrieval STructures) [20, 23] has simulated data on human memory in a number of domains including expert behaviour in board games, problem solving in physics, first language acquisition, and implicit learning.

We extend CHREST to support the acquisition and retrieval of categories, and compare its performance against a statistical learning algorithm, the support-vector machine [9, 10]. In spite of the difference between the two algorithms, one using psychologically-plausible mechanisms and the other statistical optimisation techniques, the overall performance is very similar. We also analyse the trade off between learning an index for efficient retrieval of chunks against learning the content of those chunks.

2 Chunking for Expert Performance

2.1 Chunking and Template Theories

There is considerable evidence that much of the information we learn is stored as chunks – perceptual patterns that can be used as units and that also constitute units of meaning [20]. Perhaps the best evidence for this claim comes from the study of experts (individuals that vastly outperform others in a specific domain), and in particular chess experts. In their seminal study, Chase and Simon [6] showed a chess position for a few seconds, and then asked players to reconstruct it. The analysis of the latencies between the placements of the pieces, as well as of the patterns of relations between pairs of pieces, provided clear support for the idea of chunking. Additional support came from experiments where players had to copy a given position on another board [6], to sort positions [15], and to recall positions presented either randomly, by rows and columns, or by chunks [14]. The notion of chunk also in part explains why experts can remember a considerable amount of briefly-presented material, in spite of the limits of their short-term memory [6, 11, 32].

One limit of Chase and Simon's chunking theory is that it assumes that all knowledge is stored in relatively small units (maximum of 5-6 pieces in chess). In fact, there is substantial evidence that experts also encode information using larger representations. This evidence includes verbal protocols in recall and problem solving experiments [12], and experiments where the material to recall exceeds the amount

of material that can be stored in short-term memory [22]. As the capacity of visual short-term memory is limited to about four chunks [18], the chunking theory predicts that no more than one board of about 24 pieces could be recalled. However, Gobet and Simon [22] showed that strong masters can recall up to five positions. To correct this as well as other weaknesses of the chunking theory, while keeping its strengths, Gobet and Simon developed the template theory. The key contribution of the theory, from the point of view of knowledge representation, is that it assumes that some chunks, which are met often when the agent deals with the environment, evolve into more complex data structures: templates. Templates encode both fixed information (the core) and variable information (the slots). Template theory, unlike most schema theories, proposes specific mechanisms by which templates are constructed incrementally when interacting with the environment [23]. While it takes a long time to learn a new chunk (8 seconds), information can be encoded rapidly in the slots of a template (250 milliseconds). Together with the notion of chunks, this explains experts' superiority in memory experiments with material taken from their domain of expertise, in particular when presentation times are brief.

2.2 Importance of specific perceptual knowledge

An important conclusion of research into expertise is that chunks encode specific and "concrete" information, as opposed to general and abstract information. While this is less the case with templates, which have slots to encode variable information, it is still the case that the scope of application of templates is limited by the information stored in their core, which is used to identify them in long-term memory. Several researchers have objected to this idea that chunks are specific e.g. [30], arguing that this is not efficient computationally – that is, abstract chunks would be more economical. However, the empirical evidence clearly supports the specificity hypothesis e.g. [3, 21, 34]. One example will suffice.

Bilalić [3] studied chess experts specialised in two different opening systems[1] with Black (the French and the Sicilian). When confronted with positions from the opening they did not play (e.g. "French" players with Sicilian positions), players saw their performance drop by one entire standard deviation in skill, compared to the case where they had to deal with the opening they were familiar with (e.g. "French" players with French positions). One standard deviation is a huge difference in skill. For example, when confronted with an opening she is not specialised in, the performance of a grandmaster would be similar to that of an international master.

[1] An opening system is a typical pattern for the first dozen or so moves of a chess game. Expert players must study opening systems in depth to compete successfully, and usually master five or six such systems with each colour [8].

3 CHREST

In this section, we summarise those aspects of CHREST which relate to information learning and retrieval. More detailed descriptions of CHREST are available in [12, 20]. Although we focus on chess, CHREST's mechanisms have been shown to generalise to other board games such as Awalé [17] and Go [4], as well as other domains such as language acquisition [13, 24]. An implementation of CHREST is available at `http://chrest.info`.

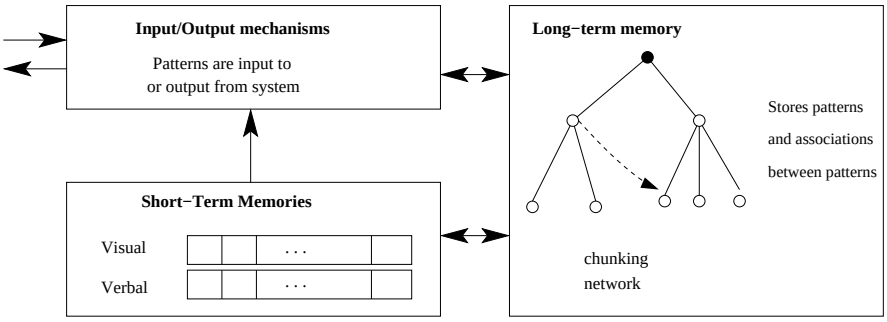


Fig. 1 CHREST cognitive architecture

3.1 Overview

Fig. 1 shows the main components of the CHREST architecture. These include input/output, a long-term memory (LTM) and a short-term meory (STM). The STM holds the current working *representation* of the stimuli being perceived; this representation is a set of pointers into LTM. STM has a limited capacity – the typical size is 4 chunks, for visual information. Each chunk is a pointer to an item held in LTM, and this item can contain varying amounts of information. The STM is populated with chunks as the eye moves over the stimulus. As the eye perceives a set of pieces on a chess board, it sorts those pieces through an index, and places the retrieved chunk into STM. The information in STM is then available for further learning, problem solving, or other cognitive processing. An example of the representation and how it is retrieved can be found in the next section.

The index into LTM is represented as a discrimination network; Fig. 3 gives an example. Nodes within LTM are connected with *test links*, each test link containing one or more primitives from the target domain which must be matched for a sorted pattern to pass down the test link. Each node contains a pattern, its *image*; these node images form the chunks and templates stored within the network. This network is efficient, in the sense that it can rapidly index sufficient quantities of information

to support expert performance [25]. CHREST supports multiple modalities (kinds of input), and can form links between nodes of different types, such as visual and verbal information. The STM contains separate lists of chunks for visual and verbal information, and the root of the LTM first sorts patterns into the visual or verbal networks depending on their modality.

3.2 Learning and Representing Chunks

As with any modelling experiment, we begin with some assumptions about what a person knows. When modelling chess players, we assume that the player can recognise an individual chess piece and their location. The 'primitives' for the representation are known as 'item-on-square', and are represented as `[P 3 4]`, denoting the name of the piece, the row and column number.[2] Fig. 2 gives an example position. When the eye perceives the board, it sees any piece it is looking at, as well as any pieces in its periphery; the size of the periphery can be varied, but in our chess experiments the eye is assumed to perceive a 5×5 grid of the chess board. Thus, looking at square (3, 7) of the board in Fig. 2, the eye would retrieve the list of pieces `<[Q 1 6] [N 4 6] [P 4 5] [P 5 6] [P 1 7] [P 2 7]>`.

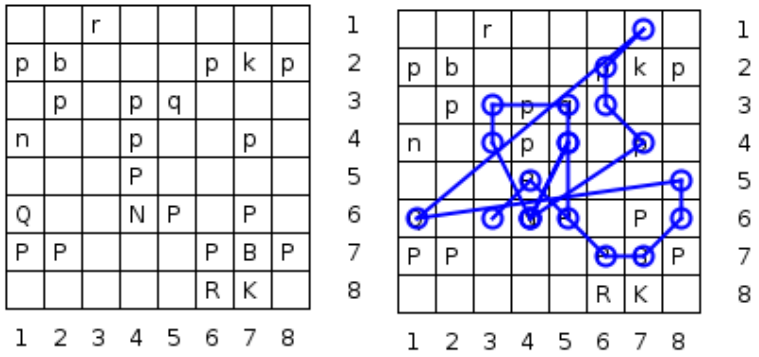


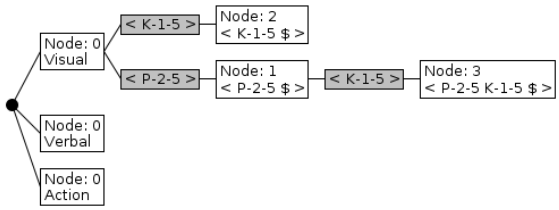
Fig. 2 Example chess position (left) and sequence of eye fixations (right)

The list of pieces is known as a 'pattern'. The perceived pattern is sorted through the model's discrimination network by checking for the presence of pieces on the network's test links. When sorting stops, the node reached is compared with the perceived pattern to determine if further learning should occur. If the perceived pattern contains everything in the retrieved pattern and some more, then *familiarisation* occurs to add some more information to the stored chunk. If the perceived pattern

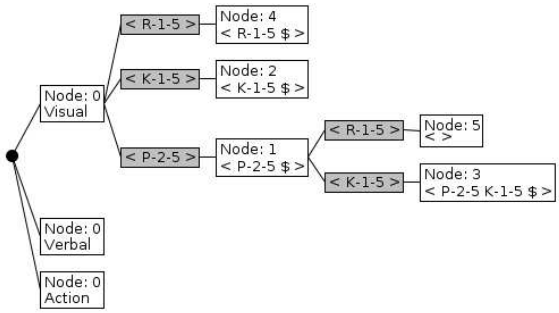
[2] Although the item-on-square representation has its limitations, it is sufficient for modelling global properties of the chess board, see [27] for a discussion.

contains some contradicting information to the retrieved pattern (such as a piece on a different square), then *discrimination* occurs, to add a further test and node to the network. Thus, discrimination increases the number of distinct chunks that the model can identify, whereas familiarisation increases the amount of information that the model can retrieve about that chunk. Fig. 3 illustrates the two learning mechanisms.

(a) After learning `[P 2 5] [K 1 5]`.



(b) Discrimination after seeing `[P 2 5] [R 1 5] [K 1 7]`



(c) Familiarising the pattern `[P 2 5] [R 1 5] [K 1 7]`

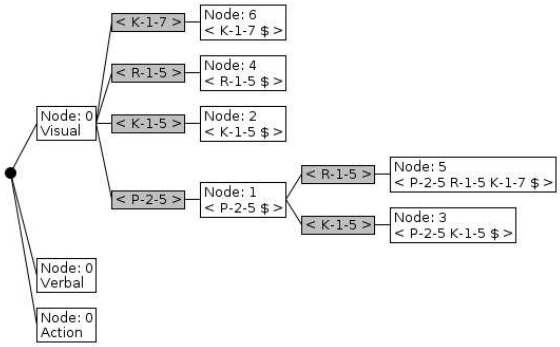


Fig. 3 Illustration of familiarisation/discrimination learning process

3.3 Eye Fixations

CHREST combines the identification and learning of chunks with an active attention mechanism, for guiding its eye about the chess board. Essentially, the implicit goal in perception is to locate the largest chunks in LTM which represent the given stimulus. Once a chunk is identified and placed in STM, the location of a piece on one of the test links is used as the next fixation point; in this way, the LTM 'drives' the eye fixations to locations likely to contain information to recognise a chunk deeper in the LTM index, and hence containing more information.

```
Fixations:
   (3, 6) Random place heuristic
   (4, 5) Random item heuristic
   (6, 7) Random item heuristic
   (7, 7) LTM heuristic
   (8, 6) Proposed movement heuristic
   (8, 5) Random place heuristic
   (1, 6) Random place heuristic
   (7, 1) LTM heuristic
   (6, 2) Random item heuristic
   (6, 3) Proposed movement heuristic
   (7, 4) Random item heuristic
   (4, 6) LTM heuristic
   (5, 4) Proposed movement heuristic
   (4, 6) LTM heuristic
   (5, 4) Proposed movement heuristic
   (4, 6) LTM heuristic
   (3, 4) Proposed movement heuristic
   (3, 3) LTM heuristic
   (5, 3) Random place heuristic
   (5, 6) Proposed movement heuristic
Chunks used:
   Node: 3371 < [P 4 5] [P 5 6] [P 6 7]
                [P 7 6] [P 8 7] >
   Node: 5780 < [P 1 7] [P 2 7] >
   Node: 435 < [P 4 5] [p 2 3] [p 4 4] [b 3 3] >
   Node: 788 < [P 5 6] [P 6 7] [P 7 6] [P 8 7]
               [K 7 8] [B 7 7] [b 7 5] [N 4 7] >
     Template:
        filled item slots:
         [R 6 8]
```

Fig. 4 An illustration of the eye fixations made by the model

As an illustration of how the eye fixations work, a model of chess positions was trained from a set of 8,644 positions, training continuing until the model's LTM contained 10,000 chunks, with each position scanned for 20 fixations. An indication of the fixations is shown in Fig. 2 (right). Fig. 4 shows a trace of the model's eye fixations on the position in Fig. 2, where the coordinates on each line refer to the

new location, and the description the reason for the move to that location. (Detailed analysis of the fit between the model's attention mechanisms and human behaviour is available [12].)

The eye fixations show the range of heuristics used in retrieving information: the eye initially is located at a random central square, and then proceeds using one of a set of heuristics. In this example, four heuristics are illustrated.

1. *Random item* fixates a previously unseen item within the field of view, but not at the centre.
2. *Random place* fixates a previously unseen square within the field of view, but not the centre.
3. *LTM* uses the largest chunk currently in STM to guide the next fixation. The heuristic proposes the location of information required to pass one of its test links.
4. *Proposed move* uses knowledge of how chess pieces move to guide the eye to objects of attack, even if located on the far side of the board.

At the end of its cycle of fixations, the model's STM contains pointers to nodes 3371, 5780, 435 and 788. The trace includes the images of these retrieved nodes; note that the last chunk listed was a template, and has one slot filled. The information in these four chunks and their links is the representation of the current stimulus.

3.4 Learning and retrieving categorisations

We now extend CHREST's mechanisms to include the acquisition of interpretations of chess positions. The technique we use is based on that for learning links across modalities [28]. The idea is that information about the recognised chunks is used to determine the position's category. Being able to categorise positions is important for chess players, as this provides crucial information about the kind of plans to follow and moves to play. In particular, knowing the opening the position comes from is indicative of the strategic and tactical themes of the board position.

As each position is scanned, a count is maintained for each chunk determining how often that chunk is associated with the position's category: this count forms the *weighting* of the chunk to that category. For example, in the experiment described below the categories are the two openings, the French and Sicilian. Each chunk will have a count of how often it has been associated with each of these two openings. Smaller chunks (such as a pair of pawns) frequently occur in many categories, but other chunks are more indicative of particular openings (such as the line of central pawns characteristic of the French defence).

For categorising a given position, the model records how many chunks it perceives which are indicative of a given category. If there is a preponderance of chunks for one category, then that is the category for the position. If there is a tie, then the weightings for each chunk's association to a category are totalled, and the highest weighting wins.

4 Experiment in Categorising Chess Positions

Our experiment investigates how our model can learn to separate chess positions into their respective opening. A dataset of chess positions from two openings was created. The dataset contains 250 positions from the French defence and 250 positions from the Sicilian defence. We run a categorisation experiment on these data with a support-vector machine, to establish a base-line performance, and with CHREST.

As the test datasets may have an imbalanced distribution of classes, the geometric mean of accuracy is used as a performance measure: if the accuracy on the two individual classes is a_1 and a_2, then the geometric mean is $\sqrt{(a_1 \times a_2)}$. This measure has the advantage of treating the two classes equally, and severely penalising poor performance in any one [26].

4.1 Support Vector Machine

For the support-vector machine experiment, the chess data were converted into features representing the 64-squares on the chess-board as a 64-element vector. Each element of the vector holds a number representing the piece at that location: 0 for empty, 1 for white pawn, up to 12 for black king. These numbers were then scaled into the range $[0,1]$, and a 0/1 class label applied representing if the position was from the French or Sicilian opening.

A support-vector machine model was created using the libsvm library [5]. The data were split randomly into 70% training and 30% testing. The training data were further split 4:1 for cross validation. A grid search for a radial-basis function kernel was performed over costs $\{2^{-5}, 2^{-3}, 2^{-1}, 2^{0}, 2^{1}, 2^{3}, 2^{5}, 2^{8}, 2^{10}, 2^{13}, 2^{15}\}$ and gammas $\{2^{-15}, 2^{-12}, 2^{-8}, 2^{-5}, 2^{-3}, 2^{-1}, 2^{1}, 2^{3}, 2^{5}, 2^{7}, 2^{9}\}$.

The best model had 112 support vectors and obtained a geometric mean of performance of 0.80 on the held-out test set.

4.2 CHREST

The dataset was split, randomly, into 70% for training and 30% for testing. A model was trained using the training data. Each position was scanned for 20 fixations, with the model learning both its discrimination network and weighted links to categories.

The model was then tested on the held-out test data. Each position was scanned for 20 fixations and the weighted categorisations retrieved for all of the perceived chunks to retrieve a category for the scanned position, as described above. This train/test process was repeated 10 times for each number of training cycles. The number of training cycles was varied from 1 to 40.

Fig. 5 shows the average over the 10 runs of the geometric mean of performance on the held-out test set, with error bars indicating ± 1 standard deviation. The per-

formance quickly reaches and maintains a level between 0.75 and 0.80, which is equivalent to the performance of the support-vector machine model.

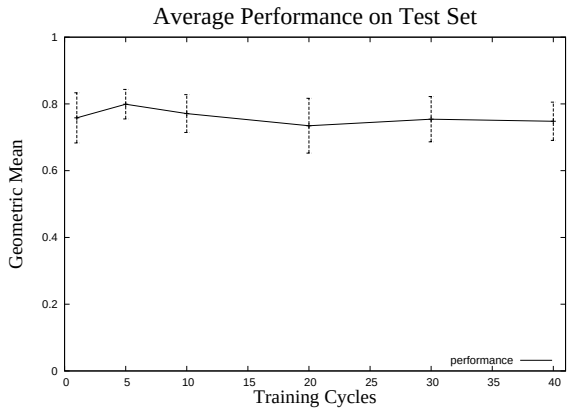


Fig. 5 Average geometric-mean performance against training cycles.

4.3 Specialisation in long-term memory

The long-term memory network grows both in size (the number of nodes) and in content (the size of the chunks stored in the nodes).

Fig. 6 shows a typical growth in the network size over time. The x-axis represents increasing number of training cycles, and the y-axis the average number of chunks within the network. The points are averaged from ten runs of the model, and error bars to ± 1 s.d. are drawn. The graph shows a growth in network size over time, with increasing variation between models as the network gets larger. The rate of growth slows with increasing training cycles, as the model becomes familiar with the training data.

Fig. 7 shows a typical growth in the network content over time. The x-axis represents increasing number of training cycles, and the y-axis the average number of pieces. The solid line is the average size of chunks stored in all the nodes of the LTM, and the dashed line is the average depth of the nodes in the network. The depth approximates the amount of information required to *retrieve* a chunk, whereas the content gives the amount of information *stored* as a chunk. The points are averaged from ten runs of the model, and error bars to ± 1 s.d. are drawn, but are relatively small (less than 0.5%).

The interesting aspect of this graph is the shape of the two lines. The line for the average depth increases sharply initially, as CHREST learns the general structure of the data. After about 5 passes through the data, this curve begins to flatten out,

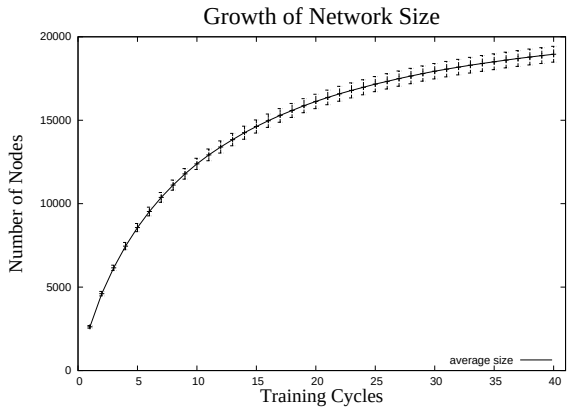


Fig. 6 Typical increase in average network size.

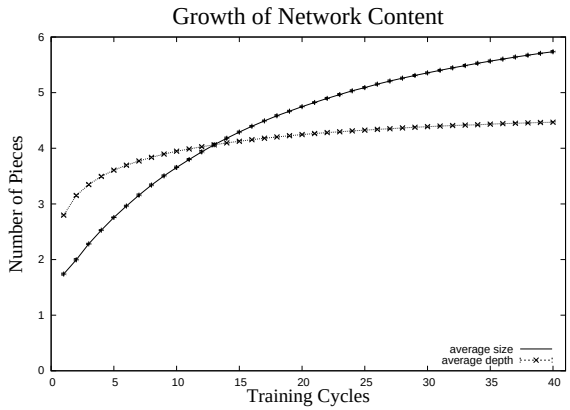


Fig. 7 Typical increase in average network depth and average chunk size against training cycles.

with an average depth of approximately 4. This indicates that CHREST has learnt the broad structure of the data, and correlates well with the change in rate of growth shown in Fig. 6.

The line for the average chunk size shows a more gradual increase over time, as the model acquires familiarity with the domain. There is a cross-over point at 14 training cycles when the average chunk size becomes larger than the average chunk depth. This indicates that CHREST is beginning to retrieve more information than it needs to perceive, and so is increasingly able to predict the structure of a position from relatively small cues. For the largest network, CHREST will typically retrieve 50% more information than it requires to access that information.

5 Conclusion

We have discussed how human performance in the game of chess presents a challenge to theories of artificial intelligence: how do humans acquire and efficiently index a large store of knowledge for efficient problem solving? We have presented the CHREST model of human perception and learning, and extended it to handle a task of categorising chess positions based on the opening they come from. Experimental results demonstrate that the the process of learning chunks enables CHREST to categorise positions with a similar reliability to a statistical learning algorithm. Analysis of the network over time illustrates the trade off in learning how to *retrieve* chunks against learning the *content* of those chunks.

In future work, we will extend this model of categorisation to support more complex interpretations of chess positions, interpretations to support quality game playing with a minimum of look-ahead search.

References

1. Anderson, J.R.: The Architecture of Cognition. Cambridge, MA: Harvard University Press (1983)
2. Berliner, H.J.: Search vs. knowledge: An analysis from the domain of games. Tech. Rep. No. CMU-CS-82-104. Pittsburgh: Carnegie Mellon University, Computer Science Department (1981)
3. Bilalić, M., McLeod, P., Gobet, F.: Specialization effect and its influence on memory and problem solving in expert chess players. Cognitive Science **33**, 1117–1143 (2009)
4. Bossomaier, T., Traish, J., Gobet, F., Lane, P.: Neuro-cognitive model of move location in the game of Go. In: Proceedings of the 2012 International Joint Conference on Neural Networks (2012)
5. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
6. Chase, W.G., Simon, H.A.: The mind's eye in chess. In: W.G. Chase (ed.) Visual Information Processing, pp. 215–281. New York: Academic Press (1973)
7. Chaslot, G., Bakkes, S., Szita, I., Sponck, P.: Monte-carlo tree search: A new framework for game AI. In: Proceedings of the Twentieth Belgian-Dutch Artificial Intelligence Conference, pp. 389–390 (2008)
8. Chassy, P., Gobet, F.: Measuring chess experts' single-use sequence knowledge using departure from theoretical openings: An archival study. PLos ONE **6** (2011)
9. Christianini, N., Shawe-Taylor, J.: An introduction to Support Vector Machines. The Press Syndicate of the University of Cambridge (2000)
10. Cortes, C., Vapnik, V.N.: Support-vector networks. Machine Learning **20**, 273–297 (1995)
11. de Groot, A.D.: Thought and Choice in Chess (First edition in 1946). The Hague: Mouton (1978)
12. de Groot, A.D., Gobet, F.: Perception and Memory in Chess: Heuristics of the Professional Eye. Assen: Van Gorcum (1996)
13. Freudenthal, D., Pine, J.M., Gobet, F.: Simulating the referential properties of Dutch, German and English root infinitives in MOSAIC. Language Learning and Development **15**, 1–29 (2009)
14. Frey, P.W., Adesman, P.: Recall memory for visually presented chess positions. Memory and Cognition **4**, 541–547 (1976)

15. Freyhoff, H., Gruber, H., Ziegler, A.: Expertise and hierarchical knowledge representation in chess. Psychological Research **54**, 32–37 (1992)
16. Gardner, H.: The Mind's New Science: A History of the Cognitive Revolution. New York: Basic Books (1987)
17. Gobet, F.: Using a cognitive architecture for addressing the question of cognitive universals in cross-cultural psychology: The example of awalé. Journal of Cross-Cultural Psychology **40**, 627–648 (2009)
18. Gobet, F., Clarkson, G.: Chunks in expert memory: Evidence for the magical number four... or is it two? Memory **12**, 732–47 (2004)
19. Gobet, F., de Voogt, A., Retschitzki, J.: Moves in Mind: The Psychology of Board Games. Hove and New York: Psychology Press (2004)
20. Gobet, F., Lane, P.C.R., Croker, S.J., Cheng, P.C.H., Jones, G., Oliver, I., Pine, J.M.: Chunking mechanisms in human learning. Trends in Cognitive Sciences **5**, 236–243 (2001)
21. Gobet, F., Simon, H.A.: Recall of random and distorted positions: Implications for the theory of expertise. Memory & Cognition **24**, 493–503 (1996)
22. Gobet, F., Simon, H.A.: Templates in chess memory: A mechanism for recalling several boards. Cognitive Psychology **31**, 1–40 (1996)
23. Gobet, F., Simon, H.A.: Five seconds or sixty? Presentation time in expert memory. Cognitive Science **24**, 651–82 (2000)
24. Jones, G.A., Gobet, F., Pine, J.M.: Linking working memory and long-term memory: A computational model of the learning of new words. Developmental Science **10**, 853–873 (2007)
25. Lane, P.C.R., Cheng, P.C.H., Gobet, F.: Learning perceptual schemas to avoid the utility problem. In: M. Bramer, A. Macintosh, F. Coenen (eds.) Research and Development in Intelligent Systems XVI: Proceedings of ES99, the Nineteenth SGES International Conference on Knowledge-Based Systems and Applied Artificial Intelligence, pp. 72–82. Cambridge, UK: Springer-Verlag (1999)
26. Lane, P.C.R., Clarke, D., Hender, P.: On developing robust models for favourability analysis: Model choice, feature sets and imbalanced data. Decision Support Systems (in press)
27. Lane, P.C.R., Gobet, F.: Perception in chess and beyond: Commentary on Linhares and Freitas (2010). New Ideas in Psychology **29**, 156–61 (2011)
28. Lane, P.C.R., Sykes, A.K., Gobet, F.: Combining low-level perception with expectations in CHREST. In: F. Schmalhofer, R.M. Young, G. Katz (eds.) Proceedings of EuroCogsci, pp. 205–210. Mahwah, NJ: Lawrence Erlbaum Associates (2003)
29. Langley, P., Laird, J.E., Rogers, S.: Cognitive architectures: Research issues and challenges. Cognitive Systems Research **10**, 141–160 (2009)
30. Linhares, A., Freitas, A.E.T.A.: Questioning Chase and Simon's (1973) "Perception in chess": The "experience recognition" hypothesis. New Ideas in Psychology **28**, 64–78 (2010)
31. Michie, D.: King and rook against king: Historical background and a problem on the infinite board. In: M.R.B. Clarke (ed.) Advances in computer chess I, pp. 30–59. Edinburgh: University Press (1977)
32. Miller, G.A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological Review **63**, 81–97 (1956)
33. Newell, A., Simon, H.A.: The simulation of human thought. Mathematics Division, The RAND Corporation, P-1734 (1959)
34. Saariluoma, P.: Location coding in chess. The Quarterly Journal of Experimental Psychology **47A**, 607–630 (1994)
35. Samsonovich, A.: Toward a unified catalog of implemented cognitive architectures. In: Proceedings of the 2010 Conference on Biologically Inspired Cognitive Architectures, pp. 195–244. Amsterdam, The Netherlands: IOS Press (2010)
36. Shannon, C.E.: A chess-playing machine. Philosophical magazine **182**, 41–51 (1950)
37. Stewart, D.: Herbert Simon: Thinking machines. Interview conducted June 1994. Transcript available at
http://www.astralgia.com/webportfolio/omnimoment/archives/interviews/simon.html (1997)
38. Turing, A.M.: Digital computers applied to games. In: B.V. Bowden (ed.) Faster than thought, pp. 286–310. London: Pitman (1953)