Constraint-Based Parsing with Distributed Representations

Peter Blouw (pblouw@uwaterloo.ca)
Chris Eliasmith (celiasmith@uwaterloo.ca)
Centre for Theoretical Neuroscience, University of Waterloo
Waterloo, ON, Canada N2L 3G1

Abstract

The idea that optimization plays a key role in linguistic cognition is supported by an increasingly large body of research. Building on this research, we describe a new approach to parsing distributed representations via optimization over a set of soft constraints on the wellformedness of parse trees. This work extends previous research involving the use of constraint-based or "harmonic"' grammars by suggesting how parsing can be accomplished using fully distributed representations that preserve their dimensionality with arbitrary increases in structural complexity. We demonstrate that this method can be used to correctly evaluate the wellformedness of linguistic structures generated by a simple context-free grammar, and discuss a number of extensions concerning the neural implementation of the method and its application to complex parsing tasks.

Keywords: natural language processing; parsing; optimization; harmonic grammar; holographic reduced representations; semantic pointer architecture

Introduction

One of the most impressive features of human cognition is the ability to rapidly process vast numbers of linguistic expressions. To help explain this ability, many cognitive scientists adopt the view that humans possess implicit knowledge of the grammatical properties that characterize well-formed phrases and sentences. An influential approach to describing this knowledge involves postulating sets of interacting constraints that favour and penalize the co-occurrence of certain structural features in the representation of a linguistic expression (Smolensky & Legendre, 2006). Applications of this approach have resulted in a number of important insights concerning the nature of language processing in cognitive systems (Prince & Smolensky, 1997).

A significant benefit of describing grammatical knowledge in terms of violable constraints is that processes sensitive to such constraints are naturally computed in neural systems (Smolensky & Legendre, 2006; Rogers & McClelland, 2014). Given as much, an important area of research concerns the development of techniques for (a) encoding structured representations into continuous vector spaces of the sort used to describe neural systems (Plate, 2003; Smolensky, 2006; Eliasmith, 2013), and (b) defining operations on these vectors spaces that perform constraint-sensitive computations of the sort required to account for linguistic phenomena (Smolensky & Legendre, 2006). To date, there has been comparatively more success solving the first of these problems than the second.

One challenge facing efforts to connect constraint-based accounts of grammatical knowledge and vector-based accounts of neural computation concerns the mapping between constraints defined over symbols and operations defined over vectors. At a theoretical level, a technique for performing this mapping has been proposed (Smolensky, 2006), but it is primarily defined using tensor product representations that grow in dimensionality in proportion to symbol structure depth. In practice, the main existing implementation of this technique uses localist rather than distributed representations (Hale & Smolensky, 2006).[1]

Our aim is to build on this previous work by implementing a constraint-based parser that operates on fully distributed representations in a vector space of fixed dimensionality at all structural depths. There are two benefits to performing these extensions. First, the use of fully distributed representations can allow for parsing behaviour that is sensitive to graded degrees of similarity between structures; such graded sensitivity to processing constraints is arguably necessary to account for many kinds of linguistic phenomena (Rogers & McClelland, 2014). Second, the use of a fixed vector space for encoding structures of all degrees of complexity provides a natural way to account for the graceful saturation of processing capabilities as structural complexity is increased. Accounting for such saturation and related processing errors is an important goal for research on linguistic cognition.

In what follows, we first describe existing approaches to describing grammatical knowledge in terms of sets of violable constraints and briefly motivate this approach in favour of more conventional approaches that postulate a set of symbolic rewrite rules capable of generating all and only the sentences of a particular language. We then describe an existing method for computing grammatical forms using constraints and describe a novel variation of this method that can allow for the implementation of a parser that operates on distributed representations in a vector space of fixed dimension. We conclude with a discussion of the current limitations of our approach, along with some promising extensions.

Harmonic Grammars

A fairly widespread view amongst contemporary cognitive scientists is that strictly rule-based accounts of linguistic phenomena are empirically inadequate. Since the late 1980's, considerable research has been directed towards the development of probabilistic and constraint-based frameworks in which certain properties of linguistic structures are favoured in a violable manner (Rogers & McClelland, 2014). Throughout this paper, we adopt the formalism of harmonic grammar (Smolensky & Legendre, 2006), in which the wellformedness

[1] In a distributed representation, each unit of a neural network participates in the encoding of numerous representations.

of a linguistic structure is evaluated against a set of soft constraints that take the following form:

> The co-occurence of structural constituents c_i and c_j should be favoured (or disfavoured) to degree w_{ij}.

Once a constraint set of this sort has been defined, it is possible to assign a scalar measure of well-formedness to every possible structure built from a fixed inventory of constituents. Formally, the scalar measure or "harmony" value, E, is a pairwise sum over the set of constituents that make up particular structure, s:

$$H(s) = \sum_{i<j} H(c_i, c_j) \tag{1}$$

where $H(c_i, c_j)$ evaluates to w_{ij} if both c_i and c_j are present in s. Importantly, the value H assigns an ordering to the set of structures containing a set of input constituents for which a parse is being sought. The maximum element of this ordering is the structure corresponding to the optimal parse of the input. Or put more intuitively, the optimal parse is the structure containing the input constituents that minimizes the overall degree of constraint violation.

The expressive power of a grammar defined in terms of simple pairwise constraints is considerable. Hale and Smolensky (2006), for example, prove that harmonic grammars can be constructed to describe formal languages in all classes of the Chomsky hierarchy. One important caveat of this result is that the resource and processing requirements associated with these grammars are often substantial. For this reason, our goal of implementing a parser whose performance degrades gracefully with increased structural complexity is an important one.

Considering the case of a simple context-free grammar helps to illustrate how Hale and Smolensky are able to describe arbitrary formal languages with large collections of pairwise constraints. A context-free grammar, to explain, is a grammar in which each production rule takes on the following form:

$$S \rightarrow \Phi \tag{2}$$

where S refers to any non-terminal symbol and Φ refers to a set of terminal and non-terminal symbols. To convert a context-free grammar into a harmonic grammar, the grammar is first translated into Chomsky normal form, in which all production rules take one of two forms:

$$S \rightarrow AB \qquad A \rightarrow a \tag{3}$$

where A and B are non-terminal symbols, and a is a terminal symbol. Notice that each production rule in (3) adds either a binary or unary branch to a parse tree while also specifying which symbols can be placed in parent-child relationships on a given branch. This observation can be used to break each binary branching rule into two parts, each of which refers to only a pair of symbols:

$$S \rightarrow A- \qquad S \rightarrow -B \tag{4}$$

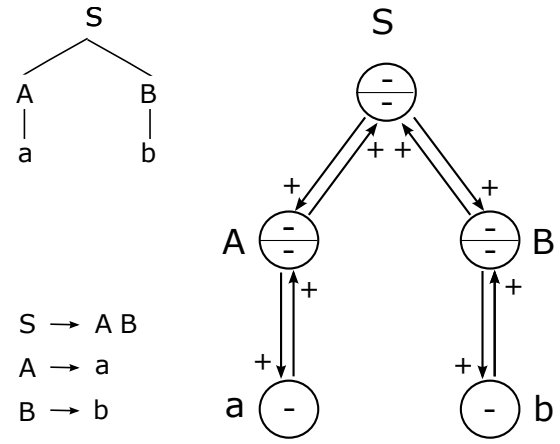

Figure 1: Harmonic grammar constraints for a small set of context-free rewrite rules, adapted from Hale and Smolensky (2006). The constraints defining the grammar are organized such that each constituent is penalized for occurring alone to a degree equal to the number of parents and children it requires.These penalties are exactly cancelled out by the positive values associated with each co-occurrence of a correct parent/child pair.

Now, it is possible to convert the rules into pairwise constraints of the form "B should be favoured as a right-child of S to degree to x" or "S should be favoured as parent to A to degree y", where x and y are derived from a constraint holding between constituents. The key to defining the grammar for a language is to organize these constraints such that the structures permitted by the language are assigned energy values of 0 via (1), while all other structures are assigned positive energy values (Hale & Smolensky, 2006). Figure 1 illustrates this constraint assignment in an intuitive manner.

Encoding Grammars in Neural Networks

At this point, it might seem that constraint-based grammars are simply complex re-descriptions of more conventional grammars. The key advantage of adopting the constraint-based approach is that it can be used to easily translate a measure of the wellformedness of a particular linguistic structure into a measure of the wellformedness in a neural system that encodes this structure. This translation occurs via a mapping from (1) to an equivalent measure of wellformedness that only refers to the state of the neural system in question.

Before describing the translation in more detail, it is necessary to briefly discuss how symbol structures can be encoded into the vectors that describe neural network states. The first step of the encoding involves performing a role decomposition on a set of symbol structures that allows each structure to be represented as a combination of role/filler pairs (Smolensky, 2006). This role decomposition defines a function $\beta : S \rightarrow \mathcal{P}(F \times R)$ that maps each symbol structure to a collection of role/filler pairs. $F \times R$ refers to the Cartesian product of the set of all possible fillers and the set of all pos-

sible roles. The roles in question are typically tree nodes, and the fillers are symbols.

The second step of the encoding involves mapping a collection of role-filler pairs into a superposition of states in a neural network. This mapping is achieved by assigning a vector to each role and filler, and by defining a binding operation that joins a role and filler into a pair. Following Plate (2003), we adopt circular convolution as binding operation and use vector addition for constructing combinations of bound items. These operations can be used to map symbol structures onto vectors as follows:

$$a = \upsilon(\beta(s)) = \sum_{F_i/R_i \in \beta(s)} F_i \circledast R_i \tag{5}$$

where a is vector, and υ indicates a mapping from a collection of role/fillers pairs onto a sum of role/filler bindings. This mapping generates what Plate refers to as a "holographic reduced representation" (HRR), which importantly is *not* a lossless encoding the of original symbol structure s.The convolution operation ensures that each role/filler binding resides in the same vector space as the vectors for each role and filler, but also it results in sums of bindings that correspond to compressed *approximations* of symbol structures.

Translating (1) into the language of HRRs is now straightforward because each structural constituent is simply a role/filler binding. As such, the wellformedness of an HRR can be assessed by summing over the constraint magnitudes associated with all pairs of role/filler bindings encoded in the HRR.

If the constraints defining a harmonic grammar are encoded into the connection weights of a neural network, then it possible to calculate the wellformedness of the network state through a simple algebraic operation that ranges over all pairwise connections in the network:

$$E(a) = -a^T W a = -\sum_{ij} a_i W_{ij} a_j \tag{6}$$

where a is an activation vector describing the state of the network, and W is the network's weight matrix. Intuitively, if two units connected by a positive weight are active, the wellformedness of the activation vector increases (i.e. fewer constraints are violated). If the weight is instead negative, the wellformedness decreases (i.e. more constraints are violated). (6) is what is often referred to as an "energy" function, and a number of results concerning the optimization of these functions in neural networks can accordingly be applied here (Smolensky & Legendre, 2006). The most important of these results entails that, assuming certain constraints on the connectivity of W, a large class of networks implement dynamics that push the activation vector a towards a steady state that is a local optimum of $E(a)$. Thus, if the weight matrix W encodes the constraints that define a harmonic grammar, and a is initialized to encode a set constituents for which a parse is being sought, then the network dynamics will shift a to a point the minimizes $E(a)$. Minima of $E(a)$, by definition, encode structures that correspond to (locally) optimal parses given that $E(a) = -H(s)$ by virtue of the mapping provided in (5).[2]

With these preliminaries out of the way, it is possible to introduce the central contribution of this paper, which is to show how to evaluate E using a network whose representational state maintains a fixed dimensionality regardless of parse complexity.

Parsing while Preserving Dimensionality

For simplicity, we initially construct our parser using the following set of context-free rewrite rules:

$$S \to A\ B \qquad A \to a$$
$$A \to C\ S \qquad B \to b \qquad C \to c$$

These abstract rules are chosen because they form a small set exhibiting the potential for boundless recursion, since an arbitrary number of branches that stem from the symbols A and S can be included in a tree. Next, we assign a random 512-dimensional unit vector to each symbol present in these rules, and to each of branch position up to some maximum depth assuming ternary branching (i.e. left, right, middle). By binding together vectors for symbols and branch positions, we can construct HRRs that encode arbitrary tree structures (Smolensky, 2006). For example, the tree in Figure 1 might be encoded as follows:

$$a = S + r_L \circledast A + r_R \circledast B + r_{LM} \circledast a + r_{RM} \circledast b \tag{7}$$

where r_L, r_R, etc. are role vectors indicating the locations of tree nodes relative to the (unlabelled) tree root.

As before, the rules defining the grammar can be reformulated as a set of pairwise constraints. The first rule states that the symbol S should have a A as its left-child and B as its right child. In the language of our encoding scheme, this means that if an HRR includes an S bound into a particular tree position, then it should also include a B bound into the next left-branching position. For each case in which one of these constraints is satisfied, the magnitude of the constraint is added to the value of $E(a)$.

We can now construct a weight matrix that assigns the proper energy values to those vectors that encode wellformed parse trees. Following Smolensky (2006), we design the weight matrix such that value of $a^T W a$ is equal to a sum of the magnitudes of all constraints that hold between pairs of constituents encoded in a. For example, if $a = S + r_L \circledast A$ and the constraint favouring A as a left child of S has a magnitude of 2, then $-a^T W a$ should equal -2. It is possible to obtain this result exactly if the vectors for all constituents in a are linearly independent (Smolensky, 2006), but our use of an HRR based encoding scheme is not guaranteed to oblige this condition. It precisely the point of this paper to show how parsers might be designed without enforcing a linear independence condition that requires the dimensionality of a to scale as a

[2]The minus sign in (6) is a convention: energy is equal to negative harmony and vice versa.

function of the number of constituents in a. We predict that as the bindings encoding different constituents become more and more dependent, the measured value of $E(a)$ will provide a progressively worse approximation to the ideal value $E(a)$ (i.e. the value calculated assuming linear independence).

To specify the exact form of W, it helps to consider the simplest case in which W encodes only a single constraint. If a encodes the two constituents used in the previous example, and the constraint holding between these constituents still has a value of 2, then there is a readily available expression in which W occurs as the only unknown quantity:

$$-2 = -(S + r_L \circledast A) \cdot W(S + r_L \circledast A) \tag{8}$$

Because matrix multiplication is linear, it is possible to break W into components that "look for" the correct parent or child of a particular constituent. For example, one component might look for an S in the position that is a parent to the position at which A is bound. (8) can therefore be split into two components that each search for a parent and child, respectively. The first of these components seeks a parent for $r_L \circledast A$, and must therefore be mapped by W_a to a vector that has an expected dot product of 1 with S and 0 with all other constituents. This result is accomplished by observing that $(S \circledast A^{-1} \circledast r_L^{-1}) \circledast (r_L \circledast A) \approx S$ and that $S \cdot S \approx 1$. As such, the matrix W_a simply needs to perform a mapping that is equivalent to convolving its input by $(S \circledast A^{-1} \circledast r_L^{-1})$. Plate (2003) demonstrates that a fixed convolution operation is equivalent to multiplication by a circulant matrix, so it is straightforward to derive W_a as a product of circulant matrices. Note that W_a maps other constituents to meaningless terms such as $(S \circledast A^{-1} \circledast r_L^{-1}) \circledast S \approx \circledast S \circledast r_L \circledast A^{-1} \circledast S$, which are treated as noise (Plate, 2003).[3]

To account for the component of W that seeks a proper child for S, the process just described is repeated to generate a matrix W_b that performs a mapping equivalent to convolving an input with $S^{-1} \circledast A \circledast r_L$. Any vector containing S is mapped by W_b to a vector that has an expected dot product of 1 with $r_L \circledast A$ and 0 with all other constituents. Adding together W_a and W_b to yields the matrix W that was originally sought to satisfy the equality described in (8), since $a^T W_a a$ and $a^T W_b a$ are both equal to one and together sum to 2 (i.e. the constraint value specified by begin with).

A final technicality concerns the fact that some of the constraints in a harmonic grammar are defined with respect to a single constituent (e.g. a tree containing A is disfavoured to degree x). This sort of constraint requires a matrix W that maps a constituent back to itself. Adding the identity matrix to W is possible way to satisfy this requirement, but doing so has the unintended consequence of reproducing the entire input to W in its output. To avoid this, we instead define a vector u that every constituent with unary constraint gets mapped to by W (different constituents are mapped to scaled versions of u to account for constraints of differing magnitudes). If u is included in the activation vector a, then each unary constraint defined over a constituent present in a will have the desired effect on $E(a)$, since $E(a) = -a^T E a$ and $-a^T W a$ will vary with the presence or absence of constituents in a that are mapped by W to u.

In the following simulations, we define a weight matrix W that is a sum of circulant matrices, each of which performs a mapping of the sort described in (8). The circulant matrices that make up W are derived by converting the simple grammar defined at the beginning of this section into a set of constraints using the methods of Hale and Smolensky (2006). Because the rewrite rule grammar produces a infinite set of trees of the form $\{c^{n-1}ab^n \mid n > 0\}$, we cap the depth of the trees under consideration at 4. This leads to a grammar that defines two trees (see Figure 2).

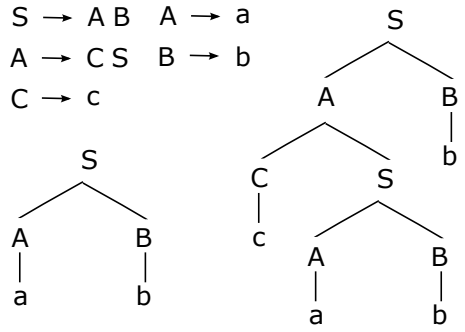

Figure 2: The two trees generated by the depth-capped grammar under consideration.

Given W and an HRR encoding some particular sum of role-filler bindings, we can compute the wellformedness of the HRR via (6). Furthermore, we can compare the wellformedness of various HRRs that encode different collections of parse tree constituents. In the simulations below, comparisons of the wellformedness of different HRRs are used to illustrate how any network that optimizes (6) will map any vector residing in a specific region of space to a new vector than encodes one of the trees defined by our grammar.

A challenge that arises when computing the wellformedness of an HRR via (6) is that the circulant matrices that comprise W introduce considerable amounts of noise into the calculation of $E(a)$. To explain, a single circulant matrix "looks for" for a particular role-filler binding in a, and if this binding is present, the circulant matrix returns a different binding (i.e. a binding that is a parent or child of the binding that was being sought). Every other constituent encoded in a is also mapped to an output by the circulant matrix, but these outputs are highly dissimilar to the role-filler bindings of interest and are therefore treated as noise. If, however, there are n circulant matrices added into W, then each each role-filler binding present in a will add $n-1$, $n-2$ or $n-3$ noise terms into the desired output of Wa (since only 1, 2, or 3 of

[3] The $^{-1}$ in these expressions denotes the pseudo-inverse of a vector with respect to convolution; a vector convolved with its pseudo-inverse is approximately equal to I, which in the context of convolution is a vector whose first element is 1 while the rest are zeros.

n matrices seeks out a given binding by virtue of the branching structure of the trees). The overall noise added to $E(a)$ has a mean of zero, but a variance that is proportional to the number of individual noise terms included in the computation of $E(a)$. To avoid this problem in our simulations, we break up the network defined by W into a set of subnetworks $\{W_1, W_2, W_3, ...W_n\}$, where n is the number of locally well-formed trees (i.e. those trees comprised of a single parent and its immediate children) permitted by the grammar, and W_n is the sum of the circulant matrices that encode the constraints corresponding to the nth local tree. Then, we break apart the vector a into a set of HRRs, each of which contains the bindings present in a that are acted on by the constraints in a given subnetwork. We compute the energy value in each subnetwork, adding together all such values to obtain the total energy of a. Put simply, we factor the overall energy function in order to compute it more accurately. With 512 dimensional vectors, the overall level noise is minimal. We use a simple thresholding operation (described below) to filter this noise. However, some unavoidable noise remains due to the fact that the vector dot product used to compute $-a_n^T W a_n$ performs pairwise comparisons between vectors that are only approximately orthogonal (see the discussion below Eq. 8)

Simulations

To illustrate the optimization behaviour that we propose to take advantage of to perform parsing, we compare the well-formedness of HRRs encoding various subsets of the set of role/filler pairs defined by our grammar. The point of this demonstration is to show that HRRs encoding the two parse trees in Figure 2 have lower energy values than all nearby HRRs. Specifically, removing a needed constituent or adding a superfluous constituent is shown to typically increase energy; from this we can conclude that an optima of the energy function $E(a)$ is obtained when is a is very close to a vector that encodes a well-formed parse tree.

In each simulation, we randomly generate vectors for all roles and fillers that can be used in the encoding of the parse trees in Figure 2. Specifically, we use random unitary vectors, which have equal exact and approximate inverses (Plate, 2003). From the random role and filler vectors, we generate a set of circulant matrices that perform linear transformations that are equivalent to convolutions (e.g. $S \circledast A^{-1} \circledast r_L^{-1}$). The circulant matrices corresponding to the constraints defining a local tree are added together and included in the set $\{W_1, W_2, W_3, ...\}$.

For a given vector a we compute $E(a)$ by first breaking apart a into subsets of bindings that each correspond to one of the subnetworks used to factor the energy function. Next, for each subnetwork, we compute $E_n(a_n)$ by multiplying a_n by W_n, the weight matrix of subnetwork n. The result of this multiplication is a vector, and we take the dot product of this vector and the vectors that the subnetwork's constraints are are designed to produce (i.e. certain child and parent bindings). The dot products are then thresholded (typically to 0 or 1) and again multiplied by the vectors the subnetwork's constraints are designed to produce, yielding "clean" versions of these vectors. This sort of clean-up function is frequently implemented in models that make use of HRRs (Eliasmith, 2013; Plate, 2003). The clean vectors are subsequently added together, and the dot product of this sum and a_n is calculated to produce a value for $E(a_n)$. In other words, we compute $E(a_n) = -a_n^T C(W_n a_n)$ in each subnetwork n, where C refers to the noise cleanup operation. $E(a)$ is computed by summing the energy values in each subnetwork.

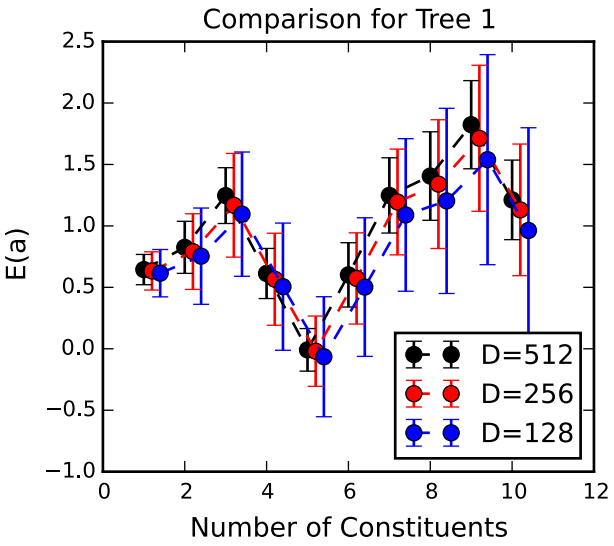

Figure 3: Comparison of well-formedness for different tree structures. The minimum in the graph corresponds to a structure that encodes the 5 constituents present in the first tree in Figure 2. Points to the left of the minimum correspond to structures that are missing constituents in the tree, while points to the right of the minimum correspond to structures that have extra constituents not found in the tree. Each plotted value is the mean of 250 trials involving randomly generated role and filer vectors of dimension D. Error bars indicate 95% confidence intervals.

Next, we perform a comparison of wellformedness across a number of linguistic structures. In the comparison shown in Figure 3, we evaluate structures that progressively include more constituents from the first tree shown in Figure 2. The wellformedness measure $E(a)$ minimizes when all five constituents of the tree are present, and increases as additional constituents are added. This result suggests that the well-formedness of a structure can be accurately tracked *even though* it only encodes a compressed approximation of the symbol structure that the wellformedness measure is defined over. In the comparison shown in Figure 4, we perform an analogous evaluation with structures that include progressively more constituents from the second tree shown in Figure 2. All simulations are run using 128, 256, and 512 dimensional vectors. As expected, performance degrades with

lower dimensional vectors.

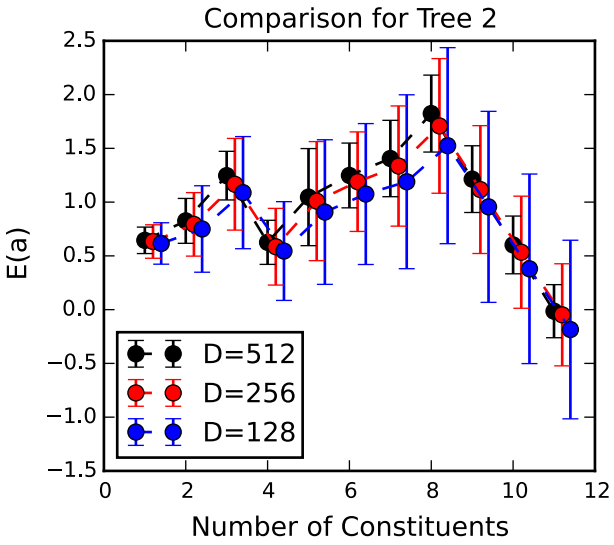

Figure 4: Comparison of well-formedness for different tree structures. The minimum in the graph corresponds to a structure that encodes the 11 constituents present in the second tree in Figure 2. Points to the left of the peak correspond to structures that are missing constituents in the tree. The second minimum likely arises as due to structures with fewer constituents having a greater degree of similarity to the first tree. Each value is the mean of 250 trials and error bars indicate 95% confidence intervals.

Overall, the key implication of these results is that a lossy encoding of a set of symbol structures into a vector space seems to preserve the properties of an energy function defined over the much larger space that is required to encode the structures exactly.

Conclusions and Future Directions

The main purpose of this work is to extend earlier research on harmonic grammars and optimization in linguistic cognition. One limitation of our current work is that we have not discussed the problem of selecting a starting state for the parsing network given an input expression (e.g. 'the dog ran'). Methods for solving this problem described by Hale and Smolensky (2006) could likely be incorporated into our framework.

It is also unknown whether our use of factoring and thresholding to eliminate noise will have any impact on whether or not a neural network can be easily designed to optimize $E(a)$. As mentioned, if filtering is used, then the expression for $E(a)$ changes slightly to $-a^T C(Wa)$ where C denotes a noise clean-up function. One approach to solving this problem involves the use of the Neural Engineering Framework (Eliasmith & Anderson, 2003), a theory that describes methods for computing arbitrary functions in networks of spiking neurons. If it is possible to define a function that optimizes $-a^T C(Wa)$, then it should also be possible to compute this function using simulated neurons. Extracting subsets of the bindings in a may also incur additional computational costs in a neural network implementation.

One other important limitation concerns the need to scale our methods to more complex trees involving large numbers of bindings. Preliminary empirical tests indicate that this is possible (e.g. using trees including on the order of 100 bindings), and theoretically, good scaling is to be expected, since the values of the noise terms introduced in the calculation of $E(a)$ are dependent on the complexity of the local trees evaluated in each subnetwork, which have at most only two direct children. Finally, on a more speculative front, it is worth noting that constraint-based parsers need not be restricted to computing syntactic transformations. In general, a parser of this sort finds a local solution to an optimization problem, so it is conceivable that many constraint-sensitive cognitive processes can be modelled using the framework proposed here. Another interesting idea concerns integrating this sort of parsing into the Semantic Pointer Architecture (Eliasmith, 2013), a recently proposed framework for describing the functional organization of neurobiological systems. Pursuing these ideas over the long term will hopefully lead to the development of scalable and robust models of constraint-based language processing.

Acknowledgments

This research was supported by CFI, OIT, SSHRC, NSERC Discovery Grant 261453, AFOSR Grant FA8655-13-1-3084, and the Canada Research Chairs program.

References

Eliasmith, C. (2013). *How to build a brain: An architecture for neurobiological cognition*. New York, NY: Oxford University Press.

Eliasmith, C., & Anderson, C. (2003). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. Cambridge, MA: MIT Press.

Hale, J., & Smolensky, P. (2006). Harmonic grammars and harmonic parsers for formal languages. In *The harmonic mind: From neural computaton to optimality-theoretic grammar* (p. 393-415). MIT Press.

Plate, T. (2003). *Holographic reduced representations*. Stanford, CA: CSLI Publications.

Prince, A., & Smolensky, P. (1997). Optimality: From neural networks to universal grammar. *Science*, *275*(5306), 1604-1610.

Rogers, T., & McClelland, J. (2014). Parallel distributed processing at 25: Further explorations in the microstructure of cognition. *Cognitive Science*, *38*(1024-1077).

Smolensky, P. (2006). Tensor product representations: Formal foundations. In *The harmonic mind: From neural computaton to optimality-theoretic grammar* (p. 271-344). MIT Press.

Smolensky, P., & Legendre, G. (2006). *The harmonic mind: From neural computaton to optimality-theoretic grammar* (Vol. 1). Cambridge, MA: MIT Press.