Distributed Representation of Syntactic Structure by Tensor Product Representation and Non-Linear Compression

Heidi H. T. Yeung   Peter W. M. Tsang

City University of Hong Kong
Tat Chee Road, Kowloon Tong,
Kowloon, Hong Kong, China

Abstract

Representing lexicons and sentences with the sub-symbolic approach (using techniques such as Self Organizing Map (SOM) or Artificial Neural Network (ANN)) is a relatively new but important research area in natural language processing. The performance of this approach however, is highly dependent on whether representations are well formed so that members within each cluster are corresponding to sentences or phrases of similar meaning. Despite the moderate success and the rapid advancement of contemporary computing power, it is still difficult to establish an efficient learning method so that natural language can be represented in a way close to the benchmark exhibited by human beings. One of the major problems is due to the general lack of effective method(s) to encapsulate semantic information into quantitative expressions or structures. In this paper, we propose to alleviate this problem with a novel technique based on Tensor Product Representation and Non-linear Compression. The method is capable of encoding sentences into distributed representations that are closely associated with the semantic contents, being more comprehensible and analyzable from the perspective of human intelligence.

Introduction

In the early days, Symbolic and Sub-symbolic approaches were generally treated as 2 separated and competing fields in the realm of Artificial Intelligence. Apparently, neither of them seemed to be capable of attaining significant breakthrough on complicated task such as natural language understanding. Prince (1997) suggested that the integration of these 2 fields could possibly achieve graceful rewards when the combined effort was focused on the principle of optimization involving language grammar and cognitive architectures. In this paper, our scope of study is formal English which is one of the most common languages tackled in the research of Natural Language Processing (*NLP*). For clarity of explanation we shall present a brief introduction on the Symbolic and Sub-symbolic processing, and techniques for integrating these two approaches.

In the early stage of study in *NLP*, it had been assumed that a sentence, in its raw form, should be sufficient for processing and analysis. The result of this simple intuitive concept was soon found to be disappointing as phrases or sentences could vary dramatically even if they conveyed identical meaning. Later, researchers discover what human interpretation of natural language was largely derived from the sentence structures rather than the exact sequence of words. It was also in line with the rule-governed models that were adopted in linguistics research for defining the inter- and intra-relations between lexicons in formal languages.

Context-free grammar (*CFG*) was often considered as a fundamental paradigm for generating complete specifications of syntax in regular languages. It involves categories (titles of words with same syntactic meaning and phrases), productions (e.g. $NP \rightarrow ART$, $NBAR$), and parsers (algorithms for generating the complete syntactic structures with a given set of productions). An effective formulation of the *CFG* in linguistic studies was given by the Chomsky Normal Form (under constrained by X-bar scheme [Chomsky 1957]) which imposed constraints in the production rules to restrict the maximum number of siblings in any non-terminal nodes. An example of applying the X-bar scheme in parsing a sentence was illustrated in Figure 1.
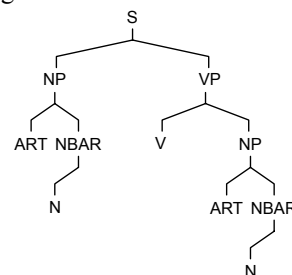

Figure 1 The parse tree of the sentence "The man feeds the dog" where "The" = ART (Article), "Dog" and "Man" = N (Noun) and "Feeds" = V (Verb)

Sub-symbolic processing was based on the idea of cognitive science in expressing concepts, patterns, sequences or structures. Instead of deriving information explicitly from the hierarchical grouping of words and phrases in a parse tree, an implicit representation was built to encapsulate sentences to certain degree of abstraction. The task could be accomplished with an encoder, often implemented as an Artificial Neural Network (*ANN*) formed by a massive network of interconnecting processors. In the investigation of sub-symbolic processing in *NLP*, Elman (1990, 1991 & 1995), Miikkulaninen (FGREP, 1993) and Farkaš (SOM on lexicon cluster, 2001) proposed some well-known researches. Conclusively, the advantages of sub-symbolism were twofold. First, the approach enabled sentences to be represented in the form of holographic memories that were implicitly recorded in the synaptic connections of a massive network. The memory established for each learned sentence was robust towards lexical variations as long as the overall sequence was not substantial changed. Second, the learning process was performed in an autonomous and black-box fashion, avoiding the meticulous analytical procedures in symbolic processing. On the downside, the approach was relatively weak in modeling syntactic structures. In addition, methods based on *ANN* also inherited all its problems such as lengthy computation time and pre-mature termination in sub-optimal states even for small set of training samples.

A solution for the first problem was proposed by Pollack (1990) with the Recursive Auto-Associative Memory (*RAAM*). In this method, a collection of sentences were first decomposed into their corresponding parse trees and encoded in the form of multidimensional vectors. The latter were taken to train up a Recurrent Neural Network (*RNN*) via repetitive epochs of synaptic weight adjustment. Upon convergence, the syntactic structures of the sample sentences were memorized in the *RNN*. Later, Kwasny (1993, 1995) had introduced a simplified version of *RAAM* known as the Sequential RAAM (*SRAAM*). A different direction was suggested by Smolensky who adopted Tensor product to represent the symbolic structures in connectionist systems [Smolensky 1990]. In his approach, a concept was encoded into a compound or multiple-rank tensor by summing up (Boolean Addition) the outer product of each constituting components v_{filter} and its corresponding role vector v_{role}, as

$$A_{Overall} = \sum v_{filter} \otimes v_{role} \qquad (1)$$

Equation (1) resulted in an expression $A_{Overall}$ known as the Tensor Product Representation (*TPR*) that encapsulated the constituents of the concept as well as their infrastructure exhibited distributed representation properties (Figure 2). The method was much faster than those based on *ANN* as iterative computation was not required. Moreover, if the role vectors are linear independent of each other the *TPR* could be perfectly reverted back to all the constituents with the Unbinding Process [Smolensky 1990].
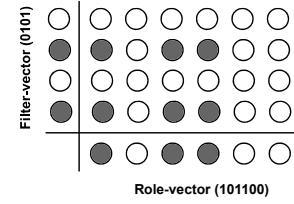

Figure 2 The tensor formed with Filler-vector (0101) and Role vector (101100)

The major limitation of this approach was that the size of the *TPR* was unbounded, directly determined by the complexity of the concept as well as the quantity of constituents. This prohibited the use of this method in natural language representation when a fixed-size expression was required for representing an unbounded syntactic structure as Figure 3.
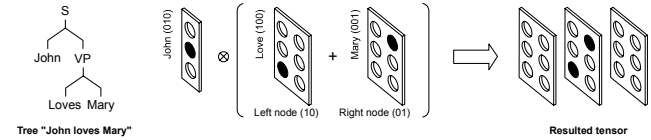

Figure 3 The tensor product representation of syntactic structure

Proposed encoding method - Tensor Product Representation with Non-linear Compression

In view of the above problems, we have developed a novel method based on Tensor Product with Non-Linear Compression so to translate sentences of varying length into fixed size representations. To start with, each unique category in a syntactic tree is mapped into a corresponding vector selected from an orthonormal basis. For clarity of description the following terminologies have been defined:

Let **T** and **G** denote the sets of tree and representation vectors, respectively. Following the Constraint Language for Attribute Trees CLAT($\mathcal{L}$) [Palm 1999], the first-order tree (e.g. Chomsky Normal Form) expression t is given by

$$t = \delta^1 \left(x_1, \pi_S^1 \left(\{ X_k \}_{k=2}^{k=c} \right) \right) \qquad (2a)$$

Sequence of children of node $x_1 = \{ X_k \}_{k=2}^{k=c} \qquad (2b)$

where x_1 represents the root node of tree t and X_k is the unexpanded k^{th} child (noted that a node may or may not contain a child) of its parent node. δ^1 is the immediate dominance of x_1 to its sequence of children, and π_s^1 is the immediate sibling precedence among the sequence of children. In addition, t may be a null collection σ if it is a termination node.

Mapping from **T** to **G** is $\xi : \mathbf{T} \to \mathbf{G}$ and the mapping function $g = \xi_T(t, \varepsilon_1)$. ε_j is the Level of Projection (*LoP*)

of jth node in the tree and $\varepsilon_1 = 1$ is the starting node for ($\forall \varepsilon_j \in \mathbf{I}^+$).

From the expression defining t and the mapping function g, we have

$$g = \xi_T(t, \varepsilon_1) = \xi_T\left(\delta^1\left(x_1, \pi_S^1\left(\{X_k\}_{k=2}^{k=c}\right)\right), \varepsilon_1\right) \tag{3}$$

The mapping function exhibited distributive properties, as

$$g = \xi_T(x_1, \varepsilon_1)\left\{\xi_T(X_k, \varepsilon_1 + 1)\right\}_{k=2}^{k=c}$$

$$g = \xi_T(x_1, \varepsilon_1)\left\{\xi_T\left(\delta^1\left(x_k, \pi_S^1\left(\{X_m\}_{m=c}^{m=c+n}\right)\right), \varepsilon_1 + 1\right)\right\}_{k=2}^{k=c} \tag{4}$$

$\xi_T(x_j, \varepsilon_j)$ contains both the information of its category and the Level of Projection of the jth node in the tree. Each category is belonged to a specific unit vector ($e_i \in \mathbf{E}^d$ for i^{th} category). The jth node in the tree is represented by its class vector $f_j \in \mathbf{E}^d$ which is an element of the set $\mathbf{E}^d$. Thus, the information encapsulated in the jth node can be expressed as $f_j = Rep(\xi_T(x_j, \varepsilon_j))$ with $\varepsilon_j = LoP(\xi_T(x_j, \varepsilon_j))$.

According to the original definition of *TPR* on syntactic structure, both nodes and branching information are deemed to be the elements contributing to the overall representation. This deviates from our approach where the *LoP* [Chomsky 1981] of the nodes is considered instead of the intra-relationships between them. To begin with, we define the set of vectors $r_k \in \mathbf{E}^{max}$ to represent the *LoP* of the tree nodes. Suppose *max* is the maximum depth of an arbitrary tree structure, we have

$$r_\varepsilon = (\tilde{r}_1\, \tilde{r}_2 \dots \tilde{r}_k \dots \tilde{r}_{max}) \text{ where } \tilde{r}_k = \begin{cases} 1 & k = \varepsilon \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Next, the mapping $\Phi : \mathbf{G} \to \mathbf{U}$ is defined for associating the concept of a node and its *LoP*. For a tree consisting of N nodes, the corresponding concepts can be concatenated through superimposition into a binary matrix given by

$$u = \Phi_G(g) = \Phi_G\left(\xi_T(x_1, \varepsilon_1)\left\{\xi_T(X_k, \varepsilon_1 + 1)\right\}_{k=2}^{k=c}\right)$$

$$u = \Phi_G(\xi_T(x_1, \varepsilon_1)) + \sum_{k=2}^{c} \Phi_G(\xi_T(X_k, \varepsilon_1 + 1)) \tag{6a}$$

where $\Phi_G(\xi_T(\sigma, \varepsilon_j)) = \mathbf{0}$ and

$$\Phi_G(\xi_T(x_j, \varepsilon_j)) = Rep(\xi_T(x_j, \varepsilon_j)) \otimes r_{LoP(\xi_T(x_j, \varepsilon_j))} = f_j \otimes r_{\varepsilon_j} \tag{6b}$$

From (6a) and (6b) a general form of the mapping $\Phi : \mathbf{G} \to \mathbf{U}$ for N number of nodes in the tree can be expressed as

$$u = \Phi_G(g) = \sum_{j=1}^{N} f_j \otimes r_{\varepsilon_j} \tag{7}$$

In the above derivations, it appears that the branching information has been neglected in building the representation. However, a closer look at (6a) reveals that the recursive expansion is in fact progressing through the nodes in a strict sequence defined by the hierarchical syntactic tree structure. The latter is derived from a parsing algorithm based on production rules and constraints governing the formation of language. For instance, the assertion of finite tree depth (non-recurrent relation), unique structure for individual sentence, and those nodes of the same category will not appear in the same *LoP*. All these imply that the branching information has been already been implicitly embedded in the representation.

The representation obtained in (7) is varying in size and difficult to be utilized by subsequent processes (a classifier, for instance) that accept fixed length inputs. To overcome this problem, we propose a novel method known as Non-linear Compression (*NLC*) for lossless compression of multiple-rank tensors to fixed-size vectors. In our method, a non-linear function ($\Theta : \mathbf{U} \to \mathbf{V}$) is defined to transform a vector into a scalar value.

In essence, suppose $a = (\tilde{a}_1\, \tilde{a}_2\, \dots\, \tilde{a}_k)$ (where $a \in \mathbf{I}^k$) is a vector to be converted and $\Omega(a, B)$ is the base-B transformation function ($B \in \mathbf{I}^+$). The transformation can result in either an integer $\Omega^+(a, B)$ or a floating-point value $\Omega^-(a, B)$. To avoid truncation error, we have chosen the latter to be the compression formula to convert an integer vector to a floating point value, as

$$\Omega^-(a, B) = \sum_{j=-max}^{-1} \tilde{a}_{max+j+1} B^j \tag{8}$$

The compression is lossless and reversible if the B^{-max} can be perfectly represented within the precision of the processor (e.g. a computer). Otherwise, the data may be subject to certain degree of distortion depending on the extent of k. The base B in the transformation is governed by the maximum discrete values that are defined within a. For binary vector, B is equals to 2.

The representation obtained in (7) can be expressed in the form of a two dimensional matrix. The row vector (fixed size) contains the category information of the tree and the column vector (variable size) encapsulates the *LoP* information. The latter is compressed into scalar value with the formula below:

$$\Theta_U(u) = \sum_{i=1}^{d} \Omega^-(u^\mathbf{T} \cdot e_i, B) \times e_i \quad \because u^\mathbf{T} = r_{\varepsilon_j} \otimes f_j \tag{9}$$

The *LoP* of each category is extracted by dot product and the Base Transformation is applied on each of them. Finally, a value contributing to specific category vector and the representation is formed by summing all contributing category vectors. The 2 mapping functions can be integrated into a general form given by the mapping function ($\Psi : \mathbf{G} \to \mathbf{V}$):

$$\Psi_G(g) = \sum_{j=1}^{N} B^{-\varepsilon_j} \cdot f_j \qquad (10)$$

Using the Figure 1 as an example, the TP Representation with NLC is as Table 1 (B = 2).

Table 1 TPR with NLC of "The man feeds the cat"

S	NP	VP	NBAR	VBAR	ART	N	V
0.5	0.375	0.25	0.1875	0.125	0.1875	0.09375	0.0625

Mathematically, an inverse transform for (9) is available as given by

$$\Phi_G(g) = \Theta_V^{-1}(\theta) = \sum_{i=1}^{d}\left(\Omega^{-1}\left(v \cdot e_i, B, \max\right) \otimes e_i\right) \qquad (11)$$

where $v = \Theta_U(u)$

Furthermore, the level of projection of the ith category in a tree can be extracted by the Unbinding Process, as

$$\Phi_U^{-1}(u,i) = [u]^{\mathbf{T}} \cdot e_i = \left[\sum_{j=1}^{N} f_j \otimes r_{\varepsilon_j}\right]^{\mathbf{T}} \cdot e_i = \sum_{j:f_j=e_i}^{N} r_{\varepsilon_j} \qquad (12)$$

According to (12), the tree can be recovered by back-tracking the nodes with reference to the parsing rules taken to construct the syntactic structures. Experimental results reveal that the full reconstruction can be achieved if the maximum depth of tree is less than 7 (cropping operation may be required for larger tree).

Experiment results

We examine the characteristics of our approach with 3 experiments. Cluster analysis is one of the important tools to explore the features inside the representation [Pollack 1990] [Kwasny 1993, 1995] by grouping candidates of similar characteristics. To evaluate the proposed method, the *TPR* of 44 sentences and phrases with different structures are computed and further encoded with *NLC*. The representations are clustered according to their Euclidean distance the result is illustrated in Figure 4.
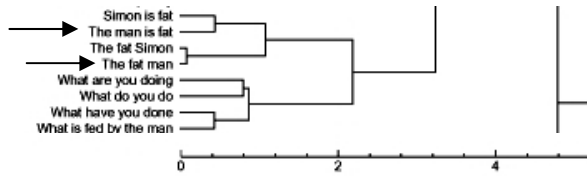

Figure 4 Cluster result of representation of 44 generated sentences

It can be seen that the sentences have been grouped according to the similarity of structures and insensitive to the actual constituents and positioning of words. For example, "The man is fat" and "The fat man" are very similar as they have three words in common. However, out method has successfully identifying them as moderately different in structure.

We have also explored the effectiveness of our method in associating representations with their semantic information. A second experiment has been conducted following the works reported by Tsang and Wong (2002). In their experiment, the *RAAM* representations were built for a set of sentences and classified into frame-based semantics (e.g. AGENT_ACTION_PATIENT) with a three-layer Back-Propagation Neural Network (*BPNN*). We repeat the test with identical set of sentences and identical settings given below (Table 2):

Table 2 Experiment parameters of second experiment

Number of grammatical categories	15
Number of semantic frame types	5
Number of Parsing Rules	18
Size of hidden layer of BPNN	12
Number of training and testing sentences	28
BPNN Training epochs	2000

Each sample sentence is presented in a shuffling manner to the *BPNN* for 100 times, each involving 2000 epochs of iterations. A plot of the average Root-Mean-Square (*RMS*) Error of M sentences and p output in BPNN against the number of iterations is plotted in Figure 5. The light and bolded lines illustrate results obtained with the representations constructed with *RAAM* and our method, respectively. It can be seen that throughout the entire training period, the *RMS* error is considerably lowered in our case except at the very beginning of the process. A very low error is finally reached (about 0.1893) with our approach whereas the system using RAAM is comparatively larger by ten times (about 1.04).
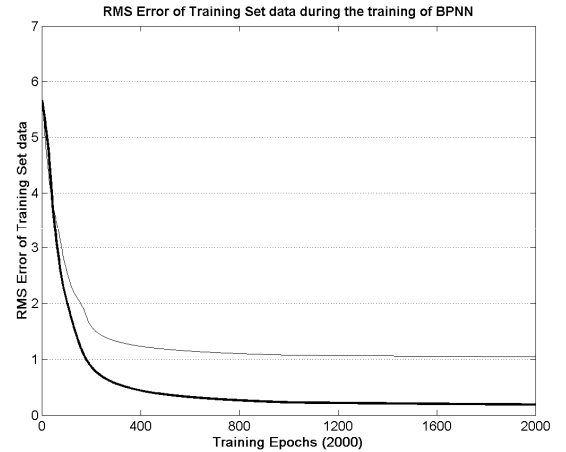

Figure 5 RMS Error of classification during training BPNN using Tsang's data set

In addition, 100% of the training and test sentences are classified properly with the use of the *NLC-TP* representation whereas only a success rate of 85.71% is recorded in Tsang and Wong's article.

The experiment is further extended to classify 14 sentences that are similar to the samples but deviated in the syntactic arrangements. Representations obtained with our method exhibit excellent *graceful degradation*. For instance, "The man sleep" and "The fat man sleep" are both classified as AGENT-ACTION although the second sentence can easily be misinterpreted as ADJ-AGENT if the first three words are considered first. The word "fat" can be considered as a kind of noise which has been successfully discarded in the *NLC-TP* representation. The difference again, is clearly reflected with the *NLC-TP* representation. However, both of the above tests failed with the *RAAM* representation.

In order to examine the representational capacities and capabilities of our approach, we design the third experiment similar to the second one but it computes with a much larger grammatical data set which can handle the more complicated linguistic structures (e.g. tenses, negation & question) and a more powerful BPNN (with doubled hidden neurons) are involved shown as Table 3.

Table 3 Experiment parameters of third experiment

Number of grammatical categories	44
Number of semantic frame types	39
Number of Parsing Rules	79
Size of hidden layer of BPNN	30
Number of sample sentences	91
Number of test sentences	50
BPNN Training epochs	2000

From the experimental results, the proposed representation seems much more compatible with BPNN than RAAM in terms of training stability and classifying correctness.

Figure 6 RMS Error of classification during training BPNN with the larger data set

We plot the average case (not the best or worst one) of RMS Error converging progress of classifying the training sentences in to assigned semantic frame type as the same way as the second experiment does in Figure 6 above. In the first half of training epochs, a significant drop of RMS Error is experienced in the training using RAAM and the error value retains around 2.27 (about 0.025 per sentence).

It is obviously trapped in a local minimum of error surface in pre-mature state. Again, the training with *NLC-TP* Representation converges in slower rates but a lower error state at 0.7176 (0.0079 per sentence) is finally reached with stable converging rates. At the mention of correctness of semantic type classification, the system with proposed representation classifies entire training set correctly but that with RAAM only achieve usually less than 70% successful rate.

Analogously, the set of testing sentences is designed for challenge the graceful degradation ability of the semantic extraction system. A difficult case is examined with the pair of sentences: "The cat would not be fed by the man" and "The cat would not be fed" that are rather similar at higher level of projection (from 5th level) but differ in the semantic frame type. The former is of type AGENT-INDIRECTACTION-PATIENT while the latter is a member of AGENT-INDIRECTACTION. The system with our representation recognizes this distinction with a clear sense (very low classification error). Nonetheless, that with RAAM representation is also failed this test as second experiment and it even cannot signify a most possible semantic type for the second sentence.

Comparing the classifying correctness on both second and third experiments, the results are generalized as Table 4a and 4b below averaging of 100 result sets. NLC-TP representation exhibits the overwhelming success in this semantic extraction application.

Table 4a Generalized results of second experiment for 100 result sets

	RAAM	NLC-TPR
Lowest Error	1.1412	0.1866
Average Error per sentence	0.0815	0.0133
Correctness of Training Set (%)	85.17	100
Correctness of Testing Set (%)	57.14	100
Computation time (in terms of t)	30 t	t

Table 4b Generalized results of third experiment for 100 result sets

	RAAM	NLC-TPR
Lowest Error	2.1607	0.6082
Average Error per sentence	0.0237	0.0067
Correctness of Training Set (%)	69.23	100
Correctness of Testing Set (%)	32	94
Computation time (in terms of t)	100 t	t

Conclusions

In this paper we have developed a novel scheme based on Tensor Product and Non-linear Compression for encoding syntactic structures of English language into fixed size vector representations. The proposed method has a number of major advantages over existing approaches based on *RAAM*. First, the computing time is significantly reduced as *ANN* is not required in encoding the sentences. Second,

the representations are stable as they are clustered according to the similarity of their corresponding syntactic structures, rather than the detail positioning or presence of individual words. Third, similar finding is obtained in the classification of representations into semantic categories through the use of Artificial Neural Network. Experimental results reflect that representations attained with our method are capable of extracting correct semantic from sentences even if they are absent in the training set. In addition considerably fewer amounts of iterations are required by the *ANN* to converge to a state with relatively lower *Mean Square Error* (*MSE*). These favorable results demonstrated the feasibility of the proposed method and its potential in real-time application on *NLP* such as Translation and Semantic Extraction.

References

Kwasny, S. C., Kalman, B. L., & Chang, N. 1993. *Distributed Patterns as Hierarchical Structures*. Proceedings of the World Congress on Neural Networks, Portland, OR, July 1993, v. II, pp. 198-201.

Kwasny, S. C., & Kalman, B. L. 1995. *Tail-recursive distributed representations and simple recurrent neural networks*. Con. Sci., 7 (1), pp. 61-80.

Chomsky, N. 1957. *Syntactic structure*. The Hague, The Netherlands: Mouton.

Chomsky, N. 1981. *Lectures on Government and Binding*. Foris Publications, Dordrecht.

Costa, F., & Frasconi, P., & Lombardo, V., & Soda, G. 2003. Towards Incremental Parsing of Natural Language Using Recursive Neural Networks. Applied Intelligence 19, pp. 9-25

Elman, J. L. 1990. *Finding structure in time*. Cognitive Science, 14, 179-211.

Elman, J. L. 1991. *Distributed representations, simple recurrent networks, and grammatical structure*. Machine Learning, vol. 7, no. 2/3, pp. 195-226.

Elman, J. L., 1995. *Language as a dynamical system. Mind as Motion: Explorations in the Dynamics of Cognition*, Eds. R. F. Port and T. van Gelder, pages 195--225. MIT Press, Cambridge, MA, (cited in pages 23, 44).

Farkas, I., & Li, P. 2001. *A Self-Organizing Neural Network Model of the Acquisition of Word Meaning*. Proceedings of the 4th Int. Conf. on Cognitive Modeling, Fairfax, VA, pp. 67-72.

Miikkulainen, R. 1993. *Subsymbolic Natural Language Processing - An integrated model of scripts, lexicon, and memory*. MIT Press, Cambridge, MA, London.

Palm, A. 1999. *The expressivity of tree languages for syntactic structures*. The Mathematics of Syntactic Structure: Trees and Their Logics Eds. H. P. Kolb; U. Monnich, Mouton-de Gruyter. The theory of syntactic domains. Technical Report 75, Department of Philosophy, University of Utrecht. pp. 113-152

Pollack, J. B. 1989. *Implications of recursive distributed representations*. Technical report, OH 43210.

Pollack, J. B. 1990. *Recursive distributed representations*. *Artificial Intelligence*, 36, pp. 77-105.

Prince, A., & Smolensky, P. 1997. *Optimality: From neural networks to Universal Grammar*. Science, 275, 1604-1610.

Smolensky, P. 1990. *Tensor product variable binding and the representation of symbolic structures in connectionist systems*. Artificial Intelligence, pp. 46, 159-216.

Smolensky, P. 1994. *Grammar-based connectionist approaches to language*. Cognitive Science. Stabler, E.P. (1994). The finite connectivity of linguistic structure. Eds. C. Clifton, L. Frazier & K. Rayner, Perspectives on sentence processing (pp. 303--336). Hillsdale, NJ: Lawrence Erlbaum Associates.

Tsang, W. M., & Wong, C. K. 2002. *Extracting Frame Based Semantics from Recursive Distributed Representation - A Connectionist Approach to NLP*. IC-AI' 02.