Inside Outside Recursive Neural Network: A Unified Framework for Compositional Semantics and Meaning in Context

Phong Le, Willem Zuidema

1 Introduction

[...]

Firstly, we ask ourselves "Which evidence is strong enough to be used for learning compositional semantics?" To answer this question, we rely on the following observation: a human being can guess the meaning of an unknown word by making use of the meaning of its context. In other words, he computes (in his brain) the meaning of the context and then use it to predict the meaning of the target unknown word (by setting some contrains to narrow down a list of possible meanings). Hence, if he correctly predicts the meaning of the unknown word, we can, at some level of belief, say that he coherends the meaning of the context. This idea is then encaptured in the following hypothesis

> *Hypothesis 1:* The agreement betwen words and contexts provides evidence for unsupervised compositional semantics learning.

Then, there are three questions need to be answered in other to implement the hypothesis

1. How to construct phrasal semantics?
2. How to construct contextual semantics?
3. How to use contextual semantics and lexical semantics to learn compositionality functions?

In the next section, we answer these questions.

2 Inside Outside Recursive Neural Network

In this section, we answer the three questions raising in Section 1.

2.1 Recursive Neural Network (RNN)

(Socher et al., 2010) answer the first question "How to construct phrasal semantics?" by Recusive Neural Network (RNN) architecture, which is used to compute continuous phrase representations. In other to see how RNN works, let's conside the following example. Assuming that there is a consitutent with parse tree $(p_2\ (p_1\ x\ y)\ z)$ as in Figure 1. We will use a neural network, which contains a weight matrix $\mathbf{W}_1$ for left children and a weight matrix $\mathbf{W}_2$ for right children, to compute parents in a bottom up manner. Firstly, we use this network to compute p_1 based on its children x and y

$$\mathbf{p}_1 = f(\mathbf{W}_1\mathbf{x} + \mathbf{W}_2\mathbf{y} + \mathbf{b}) \quad (1)$$

where $\mathbf{b}$ is a bias, f is an activation function (e.g. *tanh* or *logistic*). Then, we use the same network to compute p_2 based on its children p_1 and z

$$\mathbf{p}_2 = f(\mathbf{W}_1\mathbf{p}_1 + \mathbf{W}_2\mathbf{z} + \mathbf{b}) \quad (2)$$

This process is continued until we reach the root node. This network is trained by a gradient-based optimization method (e.g., gradient descent) where the gradient over parameters is efficiently computed thanks to the backpropagation through structure (Goller and Kuchler, 1996). Using this architecture (and its extensions), Socher and colleagues succesfully reach state-of-the-art results in syntactic parsing (Socher et al., 2013a) and sentiment analysis (Socher et al., 2013b).

In order to use this architecture in an unsupervised learning manner, (Socher et al., 2011) replace neural network in RNN by autoencoder (and hence the new architecture is called Recursive Autoencoder - RAE), which is a feedforward neural network trained by forcing output equal to input (see Figure 2). Training a RAE is therefore to minmize the sum of reconstruction errors (i.e., $\|[\mathbf{x}';\mathbf{y}'] - [\mathbf{x};\mathbf{y}]\|^2$) at all internall nodes.

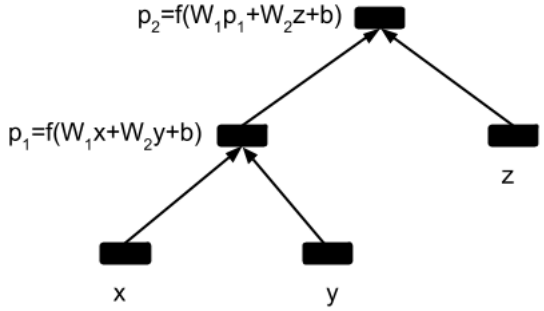

Figure 1: Recursive Neural Network (RNN).

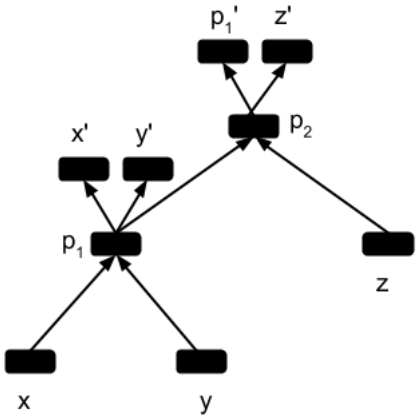

Figure 2: Recursive Autoencoder (RAE).

2.2 IORNN

None of the above architectures, RAE or RNN, compute contextual semantics. However, they give us a hint to do that. In this section, we will answer the second question "How to construct contextual semantics?" by a new neural network architecture, namely Inside Outside Recursive Neural Network (IORNN). We also present this architecture by using an example of a constituent and parse tree $(p_2\ (p_1\ x\ y)\ z)$ (see Figure 3).

Each node u is assigned two vectors $\mathbf{o}_u$ and $\mathbf{i}_u$. The first one, called *outer meaning*, denotes the meaning of the context; the second one, called *inner meaning*, denotes the meaning of the phrase that the node covers.

Word embeddings (e.g., $\mathbf{i}_x$) Similar to (Socher et al., 2010), and (Collobert et al., 2011), given a string of binary representations of words $(a, b, ..., w)$ (i.e., all of the entries of w are zero except the one corresponding to the index of the word in the dictionary), we first compute a string of vectors $(\mathbf{i}_a, ..., \mathbf{i}_w)$ representing inner meanings

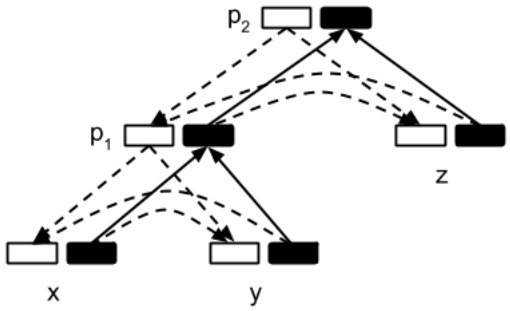

Figure 3: Inside-Outside Recursive Neural Network (IORNN). Black rectangles correspond to inner meanings, white rectangles correspond to outer meanings.

of those words by using a look-up table (i.e., word embeddings) $\mathbf{L} \in \mathbb{R}^{n \times |V|}$, where $|V|$ is the size of the vocabulary and n is the dimensionality of the vectors. This look-up table $\mathbf{L}$ could be seen as a storage of lexical semantics where each column is a vector representation of a word. Hence,

$$\mathbf{i}_w = \mathbf{L}w \in \mathbb{R}^n \tag{3}$$

Computing inner meaning The inner meaning of a non-terminal node, say p_1, is given by

$$\mathbf{i}_{p_1} = f(\mathbf{W}_1^i \mathbf{i}_x + \mathbf{W}_2^i \mathbf{i}_y + \mathbf{b}^i) \tag{4}$$

where $\mathbf{W}_1^i, \mathbf{W}_2^i$ are $n \times n$ real matrices, $\mathbf{b}^i$ is a bias vector, and $f(.)$ is an activation function, e.g. $tanh$ function. Intuitively, the inner meaning of a parent node is the function of the inner meanings of its children. This is similar to what (Socher et al., 2010) call recursive neural network.

Computing outer meaning The outer meaning of the root node, $\mathbf{o}_{root}$, is initially set randomly, and then learnt later. To a node which is not the root, say p_1, the outer meaning is given by

$$\mathbf{o}_{p_1} = g(\mathbf{W}_1^o \mathbf{o}_{p_2} + \mathbf{W}_2^o \mathbf{i}_z + \mathbf{b}^o) \tag{5}$$

where $\mathbf{W}_1^o, \mathbf{W}_2^o$ are $n \times n$ real matrices, $\mathbf{b}^o$ is a bias vector, and $g(.)$ is an activation function, e.g. $tanh$ function. Informally speaking, the outer meaning of a node (i.e., the meaning of its context) is the function of the outer meaning of its parent and the inner meaning of its sister.

The reader, if familiar with syntactic parsing, could recognizes the similarity between Equation 4, 5 and the inside, outside probabilities given a parse tree. Therefore, we name the architecture Inside-Outside Recursive Neural Network.

2.3 Training IORNN

This section is to answer the final question "How to use contextual semantics and lexical semantics to learn compositionality functions?".

According to Hypothesis 1, there must be a strong correlation between $\mathbf{o}_w$ and $\mathbf{i}_w$ where w is any word in a given sentence. The simplest way to train the network is to force $\mathbf{o}_{w_j} = \mathbf{i}_{w_j}$; hence, learning is to minimize the following loss function

$$J(\theta) = \sum_{s \in D} \sum_{w \in s} \|\mathbf{o}_w - \mathbf{i}_w\| \quad (6)$$

where D is a set of training sentences and θ are the network parameters. However, that could be problematic because the meaning of context is not necessary the meaning of the target word.

Here, based on the observation that the meaning of context sets contraints on selecting a word to fill in the blank, one could suggest put a *softmax* neuron unit on the top of each $\mathbf{o}_w$ in order to compute the probability $P(x|\mathbf{o}_w)$. Unfortunately, as pointed out by (Collobert et al., 2011), it might not work.

Using the same method proposed by (Collobert et al., 2011), we train the network such that it gives a higher score to the correct target word rather than to incorrect ones. The score $s(x, \mathbf{o}_w)$ given to a candidate word x for a specific context $\mathbf{o}_w$ is computed by

$$u(x, \mathbf{o}_w) = f(\mathbf{W}_u^o \mathbf{o}_w + \mathbf{W}_u^i \mathbf{i}_x + \mathbf{b}_u) \quad (7)$$

$$s(x, \mathbf{o}_w) = \mathbf{W}_s u(x, \mathbf{o}_w) + \mathbf{b}_s \quad (8)$$

where $\mathbf{W}_u^o, \mathbf{W}_u^i$ are $n \times k$ real matrices, $\mathbf{W}_s$ is a $k \times 1$ matrix, and $\mathbf{b}_u, \mathbf{b}_s$ are bias vectors. (We fix $k = 2n$.) Now, the objective function is the ranking criterion with respect to θ

$$J(\theta) = \sum_{s \in D} \sum_{w \in s} \sum_{x \in V} \max\{0, 1 - s(w, \mathbf{o}_w) + s(x, \mathbf{o}_w)\} \quad (9)$$

To minimize the above objective function, we randomly pick up a word in the dictionary as a corrupt example, compute the gradient, and update the parameters by a gradient descent method. Thanks to the backpropagation through structure (Goller and Kuchler, 1996), the gradient is efficiently computed. Following (Socher et al., 2013b), we use AdaGrad (Duchi et al., 2011) to update the parameters.

3 Experiments

In order to examine how IORNN performs on both compositional semantics learning and meaning in context, we evaluate it on two tasks: phrase similarity and word meaning in context. Because, to our knowledge, there are no frameworks that tackle both problems, we use vector addition and pair-wise multiplication as our baselines. Although these methods are simple, choose them as baselines are reasonable since (1) (Blacoe and Lapata, 2012) show that they perform better than RAE in the phrase similarity task, (2) they are widely used in many applications requiring compositional semantics [cite...], and (3) vector addition is widely used as a method to compute contextual meaning vector (Thater et al., 2011)

In the all experiments, we implemented IORNN in Torch-lua (Collobert et al., 2012). We initialled the network with the 50-dim word embeddings[1] from (Collobert et al., 2011). Then we trained it on a dataset containing 1.5M sentences from the BNC corpus (about one fourth of the whole corpus), which were parsed by the Berkeley parser (Petrov et al., 2006).

3.1 Phrase Similarity

Phrase similarity is the task in which one is asked to compute the meanings of (short) phrases and measure their semantic similarities. Its goodness is measured by comparing its judgements with human judgements. In this experiment, we used the dataset[2] from (Mitchell and Lapata, 2010) which contains 5832 human judgements on semantic similarity for noun-noun, verb-object, and adjective-noun phrases. There are 108 items; each contains a phrase pair and human ratings from 1 (very low similarity) to 7 (very high similarity) (see Table 1).

In this task, we use the cosine distance to measure the semantic similarity, i.e. $d(a, b) = \cos(\mathbf{i}_a, \mathbf{i}_b)$. Following (Blacoe and Lapata, 2012), (Hermann and Blunsom, 2013) and many others, we compute Spearman correlation coefficient ρ between model scores and human judgements.

First of all, we focus on the results reported by (Blacoe and Lapata, 2012). Blacoe & Lapata claims that RAE performs worse than addition and pair-wise multiplication in all of their three set-

[1] http://ronan.collobert.com/senna/

[2] http://homepages.inf.ed.ac.uk/s0453356/share

type	phrase 1	phrase 2	rating
v-obj	remember name	pass time	3
adj-n	dark eye	left arm	5
n-n	county council	town hall	4

Table 1: Items in the dataset from (Mitchell and Lapata, 2010).

tings. Here, we used their third setting, and, to be fair, we trained IORNN with their neural language model word embeddings[3]. The results are given in Table 2. IORNN is the best on adj-n and v-obj, and second on noun-noun.

dim.	model	adj-n	n-n	v-obj
50	add.	0.28	**0.26**	0.24
50	mult.	0.26	0.22	0.18
100	RAE	0.20	0.18	0.14
50	IORNN	**0.30**	0.23	**0.28**

Table 2: Spearman correlation coefficients of model predictions for the phrase similarity task with neural language model word embeddings from (Blacoe and Lapata, 2012).

(Hermann and Blunsom, 2013) extend RAE with the help of Combinatory Categorial Grammar (CCG). Their models, named Combinatory Categorial Autoencoder (CCAE), are similar to RAE but use different parameter sets for different grammatical rules and grammatical types. Thank to this semantic-related grammar, their models outperform RAE and score towards the upper end of the range of addition and pair-wise multiplication. Table 3 shows the comparison between IORNN and other methods whose results are copied from the corresponding papers.

IORNN's performance lies in the range of CCAEs on adj-n and v-obj, but worse on noun-noun. However, it is worth emphasizing that IORNN uses one parameter set for all grammatical rules (similarly to RAE). From the difference of performance between RAE and CCAEs, we expect that extending IORNN in the same way (i.e., using CCG and different parameter sets for different grammatical rules and grammatical types) will lead to better performance.

[3] http://homepages.inf.ed.ac.uk/s1066731/dl.php?file=wordVectors.emnlp2012.zip&db=1

model	adj-n	n-n	v-obj
Blacoe & Lapata			
a./m.	0.21 - 0.48	0.22 - 0.50	0.18 - 0.35
RAE	0.19 - 0.31	0.24 - 0.30	0.09 - 0.28
Hermann & Blunsom			
CCAEs	0.38 - 0.41	0.41 - 0.44	0.23 - 0.34
Our implementation			
add.	0.30	0.43	0.30
mult.	0.14	0.24	0.16
IORNN	0.38	0.36	0.32

Table 3: Spearman correlation coefficients of model predictions for the phrase similarity task.

3.2 Word Similarity in Context

Differing from the first task, this task focuses on word meaning in context: it examines how well a model can make use of context to disambiguate word meaning. In this experiment, we use the Stanford Word Similarity in Context (SWSC) dataset from (Huang et al., 2012) which contains 2003 word pairs, their sentential contexts, and human ratings from 0 to 10 (see Table 4).

For IORNN, we represent the meaning of a word in its sentential context by concatenate its inner and outer meanings, i.e. $\mathbf{m}_w = [\mathbf{i}_w; \mathbf{o}_w]$. For vector addition, we compute the context meaning by averaging the meaning vectors of 5 words on the left and 5 words on the right and then concatenate it with the meaning vector of the target word.

Similarly to the first experiment, we also use the cosine distance to measure the semantic similarity and compute Spearman correlation coefficient ρ between model scores and human judgements. Table 5 shows the comparison between IORNN and other methods.

Model	$\rho \times 100$
C&W (w/o context)	58.0
Huang et al.	
HSMN-M AvgSimC	65.7
Pruned tf-idf-M AvgSimC	60.5
Our implementation	
add.	59.0
IORNN	60.3

Table 5: ...

3.3 Summary

Now, we combine the experimental results presented above to compare IORNN with the two

word 1	word 2	human ratings
Located downtown along the east *bank* of the Des Moines River, the plaza is available for parties, ...	This is the basis of all *money* laundering, ...	0.0, 0.0, 3.0, 10.0, 8.0, 0.0, 4.0, 0.0, 0.0, 0.0

Table 4: An example in the SWSC dataset.

baselines, vector addition and vector pair-wise multiplication.

Model	Phrase similarity			WSC
	adj-n	n-n	v-obj	
add.	0.30	**0.43**	0.30	59.0
mult.	0.14	0.24	0.16	-
IORNN	**0.38**	0.36	**0.32**	**60.3**

References

Blacoe, W. and Lapata, M. (2012). A comparison of vector-based representations for semantic composition. *RAE*, 70(68.29):6910.

Collobert, R., Kavukcuoglu, K., and Farabet, C. (2012). Implementing neural networks efficiently. In *Neural Networks: Tricks of the Trade*, page 537557. Springer.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:24932537.

Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 999999:2121–2159.

Goller, C. and Kuchler, A. (1996). Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, page 347352.

Hermann, K. M. and Blunsom, P. (2013). The role of syntax in vector space models of compositional semantics. *Proceedings of ACL, Sofia, Bulgaria, August. Association for Computational Linguistics*.

Huang, E. H., Socher, R., Manning, C. D., and Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics.

Mitchell, J. and Lapata, M. (2010). Composition in distributional models of semantics. *Cognitive Science*, 34(8):13881429.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 433–440. Association for Computational Linguistics.

Socher, R., Bauer, J., Manning, C. D., and Ng, A. Y. (2013a). Parsing with compositional vector grammars. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Socher, R., Manning, C. D., and Ng, A. Y. (2010). Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., and Manning, C. D. (2011). Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics.

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013b). Recursive deep models for semantic compositionality over a sentiment treebank. EMNLP.

Thater, S., Fürstenau, H., and Pinkal, M. (2011). Word meaning in context: A simple and effective vector model. In *IJCNLP*, pages 1134–1143.