Non-Commutative Logic for Compositional Distributional Semantics

Karin Cvetko-Vah[1], Mehrnoosh Sadrzadeh[2], Dimitri Kartsaklis[3], and Benjamin Blundell[4]

[1] Faculty of Mathematics and Physics, University of Ljubljana, Slovenia
karin.cvetko@fmf.uni-lj.si
[2] School of Electronic Engineering and Computer Science, Queen Mary University of London
m.sadrzadeh@qmul.ac.uk
[3] School of Electronic Engineering and Computer Science, Queen Mary University of London
d.kartsaklis@qmul.ac.uk
[4] ITS Research, Queen Mary University of London
b.blundell@qmul.ac.uk

Abstract. Distributional models of natural language use vectors to provide a contextual foundation for meaning representation. These models rely on large quantities of real data, such as corpora of documents, and have found applications in natural language tasks, such as word similarity, disambiguation, indexing, and search. Compositional distributional models extend the distributional ones from words to phrases and sentences. Logical operators are usually treated as noise by these models and no systematic treatment is provided so far. In this paper, we show how skew lattices and their encoding in upper triangular matrices provide a logical foundation for compositional distributional models. In this setting, one can model commutative as well as non-commutative logical operations of conjunction and disjunction. We provide theoretical foundations, a case study, and experimental results for an entailment task on real data.

Keywords: non-commutative logic, compositional semantics, distributional semantics, vector semantics, skew lattices, meaning, entailment

1 Introduction

Distributional semantics is a model of natural language that works with vector representations of words embedded in a vector space of features. The vector representations are formalisations of insights of Firth and Harris [8, 10] that words that occur in similar contexts have similar meanings. These models are contrasted with traditional approaches to formal semantics where words are treated

*Karin Cvetko-Vah acknowledges the financial support from the Slovenian Research Agency (research core funding No. P1-0222). Mehrnoosh Sadrzadeh, Dimitri Kartsaklis and Benjamin Blundell acknowledge financial support from AFOSR International Scientific Collaboration Grant FA9550-14-1-0079.

as indices in a dictionary or vocabulary list, a string of letters, or the set of their denotations. The vector representations of words are built from co-occurrence matrices [23], the columns of which are features of a text, the rows of which are words, and the entries of which contain degrees of co-occurrences of the two. Vector space models provide different ways of modelling similarity relations between words [25, 21, 24], a concept that has found applications in areas such as question answering, summarisation and classification.

In [4] a mathematical framework for a unification of the distributional method and a compositional theory of grammatical types was introduced. This unification is important because the insights on which the distributional method is built mostly make sense for words. In order to obtain vector representations for phrases and sentences and to reason about their degrees of similarity, one needs to extend the distributional method from words to phrases and sentences. The unified model combines the formal grammar models of language [3, 16, 15] with the distributional theories of meaning. The result is a vector space model where the meaning of a sentence is represented by a vector computed from the vectors corresponding to the meanings of the words therein and the grammatical structure of the sentence. In [4], the two approaches are connected by the use of compact closed categories, which admit purely diagrammatic computations. These computations are related to the work by Abramsky and Coecke on the flow of information in the context of quantum information protocols [1].

Several questions were posed in [4], including extending the fragment covered there by adding their natural language coordination words to it and find proper operators corresponding to the logical connectives “and”, “or”, “but”, “unless”. This question turned out to be a challenge, since the category does not have separate products and coproducts, neither do the usual vector product and sum are fully distributive. hence they will not correspond to logical conjunction and disjunction and their variants.

In the present paper, we connect the vector model of words and sentences to skew lattice theory. Skew lattices present a non-commutative generalization of lattices, they were introduced in 1949 by Jordan [11], the author of the quantum field theory. The idea to study algebras of non-commutative idempotents arised from the realization that a pair of observables A, B corresponding to properties studied in quantum mechanics is compatible (i.e. they can be simultaneously observed) if and only if any projection corresponding to A commutes with any projection corresponding to B. The theory of skew lattices was later developed mostly by Leech, cf. [17] and [20]. The idea that the conjunction and disjunction in the natural language are sometimes non-commutative is not new. For instance, in [9] the authors argue that: “*A candle was burning on the table and the room was brightly lit* is not the same as *The room was brightly lit and the candle was burning on the table*”. ([9], p. 76–77.)

This paper is structured as follows. In Section 2, we briefly recall the case of Boolean vectors when the connectives "and", "or" are commutative. In Section 3, we generalize this setting to the non-Boolean non-commutative case. We represent our data by vectors, or more generally by matrices, so that they form a skew lattice. Then we use the skew lattice operations to represent the non-commutative connectives "and", "or". We present a case study from real data and show how entailment can be used to distinguish the non-commutative conjunction from the commutative one. We use this case study as a pilot and perform an experiment on real data. The experiment is an entailment task on a dataset of verb-object conjuncts with the two types of conjunctions. The results show that the non-commutative conjunction operator of skew lattices recognises the non-commutative conjunctive entailments better than the commutative ones.

2 Commutative connectives *and*, *or*

We assume that pieces of information are encoded as vectors. Let n be a natural number and consider the vector space $\mathbb{Z}_2^n$. The connectives *NOT, AND, OR* are encoded by the following operations:

$$\begin{array}{ll} \text{NOT} & \neg(x_1, x_2, \dots x_n) = (1 - x_1, 1 - x_2, \dots, 1 - x_n), \\ \text{AND} : & (x_1, x_2, \dots, x_n) \wedge (y_1, y_2, \dots, y_n) = (x_1 y_1, x_2 y_2, \dots, x_n y_n), \\ \text{OR} : & (x_1, x_2, \dots, x_n) \wedge (y_1, y_2, \dots, y_n) = (x_1 \circ y_1, x_2 \circ y_2, \dots, x_n \circ y_n), \end{array}$$

where $u \circ v = u + v - uv$.

It is an easy exercise to verify that $(\mathbb{Z}_2^n; \wedge, \vee)$ is a distributive lattice. In fact, it is a bounded distributive lattice with bottom $b = (0, 0, \dots, 0)$ and top $t = (1, 1, \dots, 1)$ in which every element is complemented ($\neg x$ being the complement of x). Hence $(\mathbb{Z}_2^n; \wedge, \vee, b, t)$ is a Boolean algebra.

Example 1. Assume that we measure the properties "love" and "see", like in John loves Mary *or* John sees Mary. *The information about the pair (John, Mary) is encoded by a* 4*-dimensional vector* (i, j, k, l)*, where each property is encoded by a two-dimensional vector. More precisely,* $(i, j) = (1, 0)$ *if John loves Mary and* $(i, j) = (0, 1)$ *if he hates her (which we consider to be the negation of loving her), and* $(k, l) = (1, 0)$ *if John can see Mary and* $(k, l) = (0, 1)$ *otherwise. Consider the statements:*

s_1 : John loves Mary and sees her. s_2 : John doesn't love Mary and he sees her.

What is the conjunction of statements s_1*,* s_2 *with respect to the above definition?*

$$s_1 \text{ AND } s_2 : (1, 0, 1, 0) \wedge (0, 1, 1, 0) = (0, 0, 1, 0).$$

How are we to interpret this result? Denoting $\perp = (0, 0)$ *and likewise* $\perp = (\perp, j, k) = (i, j, \perp)$ *which is interpreted by false, we obtain that* $s_1 \text{AND} s_2 = \perp$ *which sounds reasonable. Similarly, the disjunction of* s_1*,* s_2 *is obtained by:*

$$s_1 \text{ OR } s_2 : (1,0,1,0) \vee (0,1,1,0) = (1,1,1,0).$$

Denoting $\top = (1,1)$ *and* $(\top, j, k) = (j,k), (i,j,\top) = (i,j)$ *we obtain* $s_1 \vee s_2 = (1,0)$ *which corresponds to the statement* John sees Mary.

3 Non-commutative connectives *and*, *or*

There are examples in everyday life where the meaning of the connective *and* is essentially non-commutative. Consider the following sentences:

Sentence 1: *Alice found gold and ran away.*

Sentence 2: *Alice ran away and found gold.*

Although the above two sentences are both composed from the same pair of simple sentences, ie. *Alice found gold.* and *Alice ran away.* which are connected by the connective *and*, their meaning is not the same. In the case of Sentence 1, Alice most probably found gold and *then* ran away in order not to get caught or get the gold stolen from her, while in the case of Sentence 2, she first ran away from something that we are not aware of and for the reason that we don't know, and then while running away she ran into gold and found it. In the above example we saw that the connective AND as used in the natural language can have a time component implicit in it. The first action might be implicitly assumed to come before the second one, and that can effect the meaning of the sentence. There are cases in the natural language where given two actions that are connected by an AND it is natural to assume that the second one has a deeper impact. For instance, consider the following sentences:

Sentence 1: *I drank the wine and filled the glass.*

Sentence 2: *I filled the glass and drank the wine.*

While I drank a glass of wine in both cases, when the action of sentence s_1 was completed I still had a full glass, while I ended up with an empty glass when the action of sentence s_2 was completed. As we shall see below there are other instances when it is natural to glue together pieces of information by a non-commutative connective AND.

There are also situations where a non-commutative version of the connective OR is used in the natural language. To see this, consider the connective *unless* in the sentence: *Buy a blue car unless you can get a red car.* If we want to follow the above instruction, we are going to end up with either a red or a blue car, however if both colours are available, we are going to choose the red one. However, if the instruction was: *Buy a red car unless you can get a blue car*, in the case that cars of both colours were available, we would choose a blue car.

What is the right mathematical frame to encapture the above situations where the connectives AND, OR can be non-commutative? We adopt the definition below from [17].

Definition 1. *A* skew lattice *is an algebra* $(S; \wedge, \vee)$ *satisfying the following:*

- associativity *of* $\wedge$*:* $(x \wedge y) \wedge z = x \wedge (y \wedge z)$*,*
- associativity *of* $\vee$*:* $(x \vee y) \vee z = x \vee (y \vee z)$*,*
- idempotency *of* $\wedge$*:* $x \wedge x = x$*,*
- idempotency *of* $\vee$*:* $x \vee x = x$*,*
- absorptions*:* $x \vee (x \wedge y) = x = x \wedge (x \vee y)$ *and* $(x \wedge y) \vee y = y = (x \vee y) \wedge y$*.*

A skew lattice is called *strongly distributive* if it satisfies the identites

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z) \text{ and } (x \vee y) \wedge z = (x \wedge z) \vee (y \wedge z).$$

The following result is due to Leech [19] and [18].

Proposition 1. *Let* $\mathcal{P}(A, B)$ *be the set of all partial functions from* A *to* B*, for* A *and* B *non-empty sets. Given partial functions* $f, g \in \mathcal{P}(A, B)$ *we set:*

$$\text{Restriction: } f \wedge g = g|_{\mathrm{dom} f \cap \mathrm{dom} g}, \quad \text{Override: } f \vee g = f \cup g|_{\mathrm{dom} g \setminus \mathrm{dom} f}.$$

Then $(\mathcal{P}(A, B); \wedge, \vee)$ *is a strongly distributive skew lattice.*

Notice that given $f, g \in \mathcal{P}(A, B)$ as above then if $\mathrm{dom} f = \mathrm{dom}$ then $f \wedge g = g$ and $f \vee g = f$. The algebras with operations *restriction* and *override* were first studied in [2], while their connection to skew lattices was established in [7].

One more example of non-commutative conjunction, this time between adverbs: *to paint the fence white and (then) brown* vs. *to paint the fence brown and (then) white*. In the first case we end up with a brown fence, while in the second case the fence is white after we finish painting it. Again, the right one won, just like in our definition of the operation *restriction* above.

Although the connective *unless* can be seen as a non-commutative OR it is not consistent with the definition of the *override* operation above. That is because *unless* prefers the second statement, while *override* prefers the first. An intepretation of non-commutative OR that is consistent with our setting was established in [6] where the *override* operation was interpreted as $x \vee y = q(x, x, y)$, where $q(x, y, z)$ is the *Church algebra* operation satisfying the fundamental properties of the *if-then-else* connective: $q(1, x, y) = x$ and $q(0, x, y) = y$. Thus the *override* $q(x, x, y)$ can be interpreted as *if* x *then* x *else* y. In everyday language we may encounter an instance of *override* in a situation like: *Buy a blue car or else buy a red car.* Our interpretation of this sentence is: *Buy a blue car if there is one; if not, then buy a red one.* Hence, the first option is preferred.

Example 2. Assume that we observe our information by measuring the colour (or the wavelength) of an object and its size. We assign to each measurement a pair (x, y), where x is either 3 (blue), 2 (red), 1 (green) or 0 (not seen); and y is

a positive real number (in meters, for example) or 0 (not seen). Applying AND (restriction), OR (override) and interpreting 0 (not seen) as not-defined we get:

$$(3,2) \wedge (0,1) = (0,1), (3,2) \vee (0,1) = (3,2)$$
$$(0,1) \wedge (3,2) = (0,2), (0,1) \vee (3,2) = (3,1)$$

The first conjunction corresponds to (blue, 2m) AND (not seen, 1m) = (not seen, 1m), others are similarly unfolded. Notice that "blue and two meters high" is denoted by (blue, 2m), while "blue and 2m high, and not seen and 1m high" is denoted by (blue, 2m) AND (not seen, 1m). The connective AND is used for the connection between vectors, ie. pieces of full information (although some of it might be partial in that it may contain "not seen").

Another way to encode the information of Example 2 is by use of upper-triangular matrices (over the reals, for example) with 0-1 diagonals and possibly non-zero elements in the last column (0 elsewhere). Each 1 on the diagonal denotes that the property was observed (0: not observed), the element that lies in the far right column and in the row of the particular 1 denotes the value of the observed property. When we wish to encode more information we can also allow non-zero elements in the first row of the matrices.

Given the matrix ring $M_n(\mathbb{R})$ of all $n \times n$ real matrices, a subset $S \subseteq M_n(\mathbb{R})$ is called a *band* if it is closed under multiplication and $A^2 = A$ holds for all $A \in S$. Let S be such a band and let $A, B \in S$. We denote:

$$A \circ B = A + B - AB$$
$$A \nabla B = (A \circ B)^2 = A + B + BA - ABA - BAB.$$

Note that if S is closed under $\circ$ then $A \circ B = A \nabla B$ holds for all $A, B \in S$.

Proposition 2 ([5]). *Let S consist of all $(k+2)\times(k+2)$ real matrices of following form, then $(S; \cdot, \nabla)$ is a skew lattice.*

$$\begin{bmatrix} 0 & a_1 & \dots & a_k & c \\ 0 & e_1 & \dots & 0 & b_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & e_k & b_k \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix}$$

where:
(i) *each $e_i = 0$ or 1,*
(ii) *$b_i = 0$ for all i s.t. $e_i = 0$,*
(iii) *$a_i = 0$ for all i s.t. $e_i = 0$,*
(iv) *$c = a_1 b_1 + \cdots + a_k b_k$.*

Corollary 1. *Let S consist of all $(k+1) \times (k+1)$ real matrices of the following form, then $(S; \cdot, \circ)$ is a skew lattice.*

$$\begin{bmatrix} e_1 & 0 & \dots & 0 & b_1 \\ 0 & e_2 & \dots & 0 & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & e_k & b_k \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix}$$

where:
(i) *each $e_i = 0$ or 1,*
(ii) *$b_i = 0$ for all i s.t. $e_i = 0$,*

Fig. 1. Examples of commutative versus non-commutative conjunction and disjunction

$$u = \begin{bmatrix}1&0&3\\0&1&2\\0&0&0\end{bmatrix} \quad \begin{matrix}(\text{blue},2m)\ \text{AND}\ (\text{not seen},1m)\\ \begin{bmatrix}1&0&3\\0&1&2\\0&0&0\end{bmatrix}\cdot\begin{bmatrix}0&0&0\\0&1&1\\0&0&0\end{bmatrix}=\begin{bmatrix}0&0&0\\0&1&1\\0&0&0\end{bmatrix}\end{matrix} \quad \begin{matrix}(\text{blue},2m)\ \text{OR}\ (\text{not seen},1m)\\ \begin{bmatrix}1&0&3\\0&1&2\\0&0&0\end{bmatrix}\circ\begin{bmatrix}0&0&0\\0&1&1\\0&0&0\end{bmatrix}=\begin{bmatrix}1&0&3\\0&1&2\\0&0&0\end{bmatrix}\end{matrix}$$

$$v = \begin{bmatrix}0&0&0\\0&1&1\\0&0&0\end{bmatrix} \quad \begin{matrix}(\text{not seen},1m)\ \text{AND}\ (\text{blue},2m)\\ \begin{bmatrix}0&0&0\\0&1&1\\0&0&0\end{bmatrix}\cdot\begin{bmatrix}1&0&3\\0&1&2\\0&0&0\end{bmatrix}=\begin{bmatrix}0&0&0\\0&1&2\\0&0&0\end{bmatrix}\end{matrix} \quad \begin{matrix}(\text{not seen},1m)\ \text{OR}\ (\text{blue},2m)\\ \begin{bmatrix}0&0&0\\0&1&1\\0&0&0\end{bmatrix}\circ\begin{bmatrix}1&0&3\\0&1&2\\0&0&0\end{bmatrix}=\begin{bmatrix}1&0&3\\0&1&1\\0&0&0\end{bmatrix}\end{matrix}$$

Example 3. We interpret the data from Example 2 in upper triangular 3×3 matrices from Corollary 1. The first row corresponds to colour, the second one to height, and the third is zero (always, we need it so that the usual multiplication of matrices works). We assign to the vectors $(\text{blue}, 2m)$ and $(\text{not seen}, 1m)$ matrices from which we then obtain the correspondence between vectors and matrices. These are shown in Figure 1.

4 Boolean and skew semantics for natural language

For demonstration purposes, consider a simple fragment of English generated by the context free grammar of Figure 2.

Fig. 2. An exemplary context free grammar for a simple fragment of English

S	$\to$ NP VP	NP	$\to$ John, Mary, $\cdots$
VP	$\to$ V NP	Adj	$\to$ lucky, tall, red, $\cdots$
S	$\to$ S and/or S	Adv	$\to$ deeply, slowly, quickly, $\cdots$
NP	$\to$ Adj NP	VP	$\to$ sneeze, sleep,$\cdots$
VP	$\to$ VP Adv	V	$\to$ love, kiss, $\cdots$

A model for the language generated by this grammar minus the logical rule S $\to$ S and/or S, is a pair $(U, [\![\ ]\!])$, where U is universal reference set and $[\![\ ]\!]$ is an interpretation function defined by induction as follows. For terminals we have:

- The interpretation of a terminal $y \in \{np, adj, adv, vp\}$ generated by either NP $\to$ np, Adj $\to$ adj, VP $\to$ vp, Adv $\to$ adv, is $[\![y]\!] \subseteq U$. That is, noun phrases, adjectives, verb phrases, and adverbs are interpreted as unary predicates over the reference set.
- The interpretation of a terminal y generated by V $\to$ y is $[\![y]\!] \subseteq U \times U$; verbs are interpreted as binary predicates over the reference set.

For non-terminals, for all rules except for S $\to$ S and/or S, we have:

$$[\![\text{V NP}]\!] = [\![v]\!]([\![np]\!]) \quad [\![\text{NP VP}]\!] = [\![vp]\!]([\![np]\!])$$
$$[\![\text{Adj NP}]\!] = [\![adj]\!]([\![np]\!]) \quad [\![\text{VP Adv}]\!] = [\![adv]\!]([\![vp]\!])$$

Here, for $R \subseteq U \times U$ and $A \subseteq U$, by $R(A)$ we mean the forward image of R on A, that is $R(A) = \{y \mid (x, y) \in R, \text{for } x \in A\}$. To keep the notation unified, for R a unary relation $R \subseteq U$, we use the same notation and define $R(A) = \{y \mid y \in R, \text{for } x \in A\}$, i.e. $R \cap A$.

In order to interpret the logical rule S $\to$ S and/or S, we have to move to a lattice over U. If our connectives are Boolean, this lattice is $\mathcal{P}(U)$ and we have:

$$[\![\text{S and/or S}]\!] = [\![\text{S}]\!] \ \wedge / \vee \ [\![\text{S}]\!]$$

for $\wedge/\vee$ the Boolean lattice operations. In this case, we are working in a Boolean model $(\mathcal{P}(U), [\![\ ]\!])$. For non-Boolean non-commutative logical operations, we work with a skew lattice over U.

Definition 2. *A skew lattice semantics for the language generated by the grammar of Figure 2 is $(S(U); \wedge, \vee; [\![\ ]\!])$, where U is a universal reference set, $S(U)$ consists of the real matrices defined in Proposition 2 and $[\![\ ]\!]$ is an interpretation function defined by induction as follows. To terminals we assign:*

- *to each np, vp a skew matrix $[\![np]\!] := u_{np}, [\![vp]\!] := u_{vp}$ satisfying $e_1 = \cdots = e_k = 1$ and all $a_i = b_i \neq 0$,*
- *to each v a diagonal matrix $[\![v]\!] := u_v$ of the form (1) with at least one 1 on the diagonal (and all other entries 0),*
- *to each adj a skew matrix $[\![adj]\!] := u_{adj}$ of the form $e_1 = \cdots = e_k = 1$, $a_1 = \cdots = a_k \neq 0$ and $b_1 = \cdots = b_k = 0$,*
- *to each adv a skew matrix $[\![adv]\!] := u_{adv}$ of the form $e_1 = \cdots = e_k = 1$, $a_1 = \cdots = a_k = 0$ and $b_1 = \cdots = b_k \neq 0$.*

To non-terminals we assign:

- *to each x $\to$ y z, the skew matrix $[\![x]\!] = u_x := [\![y]\!] \times [\![z]\!]$,*
- *to S $\to$ S and/or S, the skew matrix $[\![S]\!] := [\![S]\!] \wedge / \vee [\![S]\!]$, for $\wedge, \vee$.*

Note that each e_i-position in the verb item corresponds to a particular verb. We call an index i a *defining index* for a skew matrix A if $a_{ii} \neq 0$. The product $A \cdot B$ can only be nonzero if A and B have at least one common defining index.

Consider the set of terminals "John, loves, sees, Mary, sleeps, lucky, deeply". We encode our data by the 5×5 matrices, presented in Figure 3. The $*$ element in u_{Mary} equals $3m^2$ and the $*$ element in u_{John} equals $3j^2$. All our matrices are idempotent, i.e. they satisfy $A^2 = A$. So we have $[\![\text{Mary}]\!] = u_{\text{Mary}}$, $[\![\text{John}]\!] = u_{\text{John}}$, $[\![\text{loves}]\!] = u_{\text{loves}}$, and $[\![\text{John loves Mary}]\!] := u_{\text{John}} \times u_{\text{loves}} \times u_{\text{Mary}}$.

Fig. 3. Example skew matrices for words

$$u_{\text{Mary}} = \begin{bmatrix} 0 & m & m & m & * \\ 0 & 1 & 0 & 0 & m \\ 0 & 0 & 1 & 0 & m \\ 0 & 0 & 0 & 1 & m \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, u_{\text{John}} = \begin{bmatrix} 0 & j & j & j & * \\ 0 & 1 & 0 & 0 & j \\ 0 & 0 & 1 & 0 & j \\ 0 & 0 & 0 & 1 & j \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, u_{\text{loves}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$u_{\text{sleeps}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, u_{\text{lucky}} = \begin{bmatrix} 0 & l & l & l & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, u_{\text{deeply}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & d \\ 0 & 0 & 1 & 0 & d \\ 0 & 0 & 0 & 1 & d \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$[\![\text{Mary loves John}]\!] = \begin{bmatrix} 0 & m & 0 & 0 & mj \\ 0 & 1 & 0 & 0 & j \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, [\![\text{Mary sleeps}]\!] = \begin{bmatrix} 0 & 0 & 0 & m & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$[\![\text{lucky Mary}]\!] = \begin{bmatrix} 0 & l & l & l & 3lm \\ 0 & 1 & 0 & 0 & m \\ 0 & 0 & 1 & 0 & m \\ 0 & 0 & 0 & 1 & m \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad [\![\text{Mary sleeps deeply}]\!] = \begin{bmatrix} 0 & 0 & 0 & m & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & d \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

5 A case study from real data

In this section, we use real data to build our matrices and detail the computations for one set of the example sentences of the paper. We perform a case study with 2 verbs, two nouns, and one adjective. We work with 5×5 matrices, where one dimension of each verb matrix is reserved such that the two verbs of the example have a common defining index.

Consider the lemmatised versions of the sentences"filled glass and drank wine" and "drank wine and filled glass" of Section 3. We fill the matrices of words of these sentences with real data and compute their conjunction, in the two presented orders. After the first conjunct, the glass will be full and after the second conjunct, the glass will be empty. We verify if this fact indeed follows from real data, by computing each entailment and observing which one of the conjuncts entails "full glass" with a larger degree.

The terminals of our sentences are "filled, drank, glass, wine, full, empty". We build skew matrices for these. The a_i, b_i entries for nouns and adjective matrices are obtained from the PPMI-normalised version (see Appendix for the PPMI formula and its explanation) of the degree of co-occurrence of each word with the two features of "full" and "empty". The entries $a_2 = b_2$ correspond to the feature "empty" and the entries $a_3 = b_3$ to the feature "full". Dimension 4

records the common defining index. We copy the information corresponding to the feature "full" in this dimension. The reason we are copying this dimension and not dimension 2 is because we are verifying the degree of entailment with the phrase "full glass" (and not with "empty glass"), thus entries $a_4 = b_4$ should be the same as $a_3 = b_3$. This is important for matrices of the verbs, where cell c_{33} records the common defining index of the "drank'" and "filled"matrix.

Note that, in a Boolean setting the two properties of being full and being empty are opposites of each other: if one of them is 1, the other will be 0. In real scenarios, however, this is not necessarily the case. For instance, i the data is presented in Figure 4, in the matrix of "glass", the PPMI-normalised versions of the number of times "glass" occurred 5 words close to features "full" and "empty" are 11 and 10.2, respectively. This is because a glass can be empty and it can be full in different contexts.

Fig. 4. A set of word matrices derived from data

$$\text{glass} = \begin{bmatrix} 0 & 10.2 & 11 & 11 & 346 \\ 0 & 1 & 0 & 0 & 10.2 \\ 0 & 0 & 1 & 0 & 11 \\ 0 & 0 & 0 & 1 & 11 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{wine} = \begin{bmatrix} 0 & 8.7 & 10.9 & 10.9 & 313.3 \\ 0 & 1 & 0 & 0 & 8.7 \\ 0 & 0 & 1 & 0 & 10.9 \\ 0 & 0 & 0 & 1 & 10.9 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{full glass} = \begin{bmatrix} 0 & 1 & 1 & 1 & 32.2 \\ 0 & 1 & 0 & 0 & 10.2 \\ 0 & 0 & 1 & 0 & 11 \\ 0 & 0 & 0 & 1 & 11 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\text{drank} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{filled} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{full} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We compose these word matrices an obtain matrices for phrases, we then form their two possible conjunctions, resulting in matrices of Figure 5. We see that "drank wine and filled glass" is closer (although very slightly so) to "full glass" than "filled glass and drank wine". This is because the entry b_3 of the first conjunct is 11, this is closer to the same entry in "full glass" that is, 11, than the b_3 of the second conjunct, which is 10.9. All the other entries are of equal distance of the entries of "full glass". The difference is small due to the fact that we took the features "full" and "empty" to be the same as the words "full" and "empty" and that these words are examples of words that do often occur in similar contexts, thus we get very similar numbers for them, i.e. 11 and 10.2. A more refined analysis on features that are not similar will reflect better on data.

Fig. 5. The set of conjunctive phrase matrices built from word matrices Figure 4.

$$\text{drank wine and filled glass} = \begin{bmatrix} 0&0&0&0&0 \\ 0&1&0&0&8.7 \\ 0&0&1&0&10.9 \\ 0&0&0&0&0 \\ 0&0&0&0&0 \end{bmatrix} \cdot \begin{bmatrix} 0&0&0&0&0 \\ 0&0&0&0&0 \\ 0&0&1&0&11 \\ 0&0&0&1&11 \\ 0&0&0&0&0 \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0 \\ 0&0&0&0&0 \\ 0&0&1&0&11 \\ 0&0&0&0&0 \\ 0&0&0&0&0 \end{bmatrix}$$

$$\text{filled glass and drank wine} = \begin{bmatrix} 0&0&0&0&0 \\ 0&0&0&0&0 \\ 0&0&1&0&11 \\ 0&0&0&1&11 \\ 0&0&0&0&0 \end{bmatrix} \cdot \begin{bmatrix} 0&0&0&0&0 \\ 0&1&0&0&8.7 \\ 0&0&1&0&10.9 \\ 0&0&0&0&0 \\ 0&0&0&0&0 \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0 \\ 0&0&0&0&0 \\ 0&0&1&0&10.9 \\ 0&0&0&0&0 \\ 0&0&0&0&0 \end{bmatrix}$$

6 Large scale entailment experiment

In previous work [13, 12], we developed theory for and experimented with entailment in compositional distributional semantics. We built three entailment datasets from real data by using linguistic resources such as WordNet. These datasets consist of subject-verb and verb-object phrases and subject-verb-object sentences. We worked with different degrees of feature inclusion on the vectors and matrices of these phrases and sentences and measured the entailments thereof based on these degrees. In this section, we repeat the experiment of the previous section on a logical extension of the verb-object part of this dataset.

Our skew matrices are 300×300 and their entries are normalised using probabilistic versions of raw co-occurrences and their non-negative logarithms, a measure known as Positive Pointwise Mutual Information (PPMI); the formulae and explanations for these are given in the Appendix. The raw co-occurrence counts (before normalisation) were collected in the context of a 5-word window around the words. The verb-object dataset has 436 verb-object pairs, 218 of which stand in a positive entailment relationship with each other and 218 in a negative one. A snapshot of the positive entailments is presented in Figure 6. The negative entries are the reverses of the positive ones. For an explanation on how these datasets are built, please see [12].

We extended the above dataset with commutative and non-commutative conjunctions in the following way. From each two entries of the dataset $vo_1 \vdash vo_2$ and $vo_1' \vdash vo_2'$, we form two conjunctive entries, of the following forms

$$vo_1 \wedge vo_1' \vdash vo_2 \wedge vo_2' \qquad \text{and} \qquad vo_1' \wedge vo_1 \vdash vo_2 \wedge vo_2'$$

We then compute a matrix for each of the vo's (i.e. for vo_1, vo_2, vo_1', vo_2'). We compute their skew conjunctions, with the goal of verifying whether this non-commutative conjunction does perform better on recognising the conjunctive

entailments of the first case above. In the first case, vo_1 entails vo_2 and vo'_1 entails vo'_2, hence $vo_1 \wedge vo'_1$ should entail $vo_2 \wedge vo'_2$. This entailment fails for second case, because the conjunction is non-commutative, that is vo'_1 does entail vo_2, similarly, vo_1 does not entail vo'_2, thus the entailment between their non-commutative conjunctions fails.

Fig. 6. Examples from the verb-object entailment dataset and results of the non-commutative conjunction experiment with the PPMI and probability ratio on the 1st sample of dataset.

$VO \vdash V'O'$
sign contract ⊢ write agreement
publish book ⊢ produce publication
sing song ⊢ perform music
reduce number ⊢ decrease amount
promote development ⊢ support event

	Inclusion	APinc	BAPinc	SAPinc	SBAPinc
PPMI					
non-comm.	0.52	0.67	0.60	0.82	0.82
comm.	0.51	0.63	0.58	0.81	0.80
Probability					
non-comm.	0.58	0.65	0.61	0.80	0.79
comm.	0.56	0.60	0.57	0.79	0.77

The results are evaluated by a binary classification of the existing entailment measures: APinc, BAPinc, SAPinc, SBAPinc. These are from the distributional literature on degrees of entailment between words and sentences, the formulae for computing them and explanations thereof are presented in the Appendix. As we are not working with Boolean models, we will have degrees of entailment and report Area Under Curve; this returns an evaluation of the entailment at every possible non-zero threshold. The baseline is labelled "Inclusion": the binary entailment between the features. Since our sample size is large (about 6000, obtained by recasting all of the conjuncts against each other), we performed the experiments on random subsets of the dataset, each with size 1000. The results of the first sample are in right hand table of Figure 6. The results of the second sample are in the Appendix.

With all of the measures and in both normalisation schemes, the non-commutative conjunction comes out as a more appropriate operation for judging the non-commutative entailments. These results are preliminary, they are based on word-word matrices. A more appropriate empirical evaluation will be obtained by working on word-feature matrices, where the columns are not just word, but a set of words clustered together using feature induction techniques such as Single Value Decomposition (SVD).

7 Conclusions and future work

We reviewed the theory of skew lattices, which formalise a logic with non-commutative conjunction and disjunction. We motivated the existence of these operations in natural language. We presented an account of compositional distributional semantics where meanings of words, phrases, and sentences are vectors. We then showed how the data represented by skew lattices is encoded in matrices and developed a skew lattice semantics for compositional distributional models. Treating logical operators has been a challenge to these models and this paper provides a solution. We related our work to real data by first recasting one of the examples of the paper against co-occurrence matrices and then performed an experiment on a conjunctive entailment task. A similar experiment can be performed for the non-commutative disjunction, this is left to future work. On the theoretical side, with the current definitions, the matrices of verbs have to have 0-1 entries. We aim to generalise the setting, either by changing the matrices of nouns and adjectives, or the definition of the non-commutative conjunction, so we can populate all the matrices with real co-occurrence counts.

References

1. Abramsky, S., Coecke, B.: A categorical semantics of quantum protocols. In: In: Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science (LiCS?04), IEEE Computer Science. Press. Arxiv:quant-ph/0402130 (2004)
2. Berendsen, J., Jansen, D.N., Schmaltz, J., Vaandrager, F.W.: The axiomatization of override and update. J. Applied Logic 8, 141–150 (2010)
3. Chomsky, N.: Three models for the description of language. IRE Transactions on Information Theory 2, 113–124 (1956)
4. Coecke, B., Sadrzadeh, M., Clark, S.: Mathematical Foundations for Distributed Compositional Model of Meaning. Lambek Festschrift. Linguistic Analysis 36, 345 – 384 (2010)
5. Cvetko-Vah, K.: Skew lattices of matrices in rings. Algebra Universalis 53, 471–479 (2005)
6. Cvetko-Vah, K., Salibra, A.: The connection of skew boolean algebras and discriminator varieties to church algebras. Algebra Universalis 73, 369–390 (2015)
7. Cvetko-Vah, K., Leech, J., Spinks, M.: Skew lattices and binary operations on functions. J. Applied Logic 11, 253–265 (2013)
8. Firth, J.: A synopsis of linguistic theory 1930–1955. In: Studies in Linguistic Analysis (1957)
9. Galatos, N., Jipsen, P., Kowalski, T., Ono, H.: Residuated Lattices: An Algebraic Glimpse at Substructural Logics, Studies in logic and the foundations of mathematics, vol. 151. Elsevier, Amsterdam (2007)
10. Harris, Z.: Distributional structure. Word (1954)
11. Jordan, P.: Über nichtkommutative verbände. Arch. Math. 2, 56–59 (1949)
12. Kartsaklis, D., Sadrzadeh, M.: A compositional distributional inclusion hypothesis. In: Logical Aspects of Computational Linguistics. Celebrating 20 Years of LACL (1996-2016) - 9th International Conference, LACL 2016, Nancy, France, December 5-7, 2016, Proceedings. Lecture Notes in Computer Science, vol. 10054, pp. 116–133. Springer (2016)
13. Kartsaklis, D., Sadrzadeh, M.: Distributional inclusion hypothesis for tensor-based composition. In: COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan. pp. 2849–2860. ACL (2016)

14. Kotlerman, L., Dagan, I., Szpektor, I., Zhitomirsky-Geffet, M.: Directional distributional similarity for lexical inference. Natural Language Engineering 16(04), 359–389 (2010)
15. Lambek, J.: Type grammars revisited. In: proceedings of LACL 97. Lecture Notes in Artificial Intelligence, vol. 1582, pp. 1–27. Springer Verlag (1997)
16. Lambek, J.: The mathematics of sentence structure. American Mathematical Monthly 65, 154–170 (1958)
17. Leech, J.: Skew lattices in rings. Algebra Universalis 26, 48–72 (1989)
18. Leech, J.: Skew boolean algebras. Algebra Universalis 27, 497–506 (1990)
19. Leech, J.: Normal skew lattices. Semigroup Forum 44, 1–8 (1992)
20. Leech, J.: Recent developments in the theory of skew lattices. Algebra Universalis 52, 7–24 (1996)
21. Lin, D.: Automatic retrieval and clustering of similar words. In: Proceedings of the 17th international conference on Computational linguistics-Volume 2. pp. 768–774. Association for Computational Linguistics (1998)
22. Lin, D.: An information-theoretic definition of similarity. In: Proceedings of the International Conference on Machine Learning. pp. 296–304 (1998)
23. Rubenstein, H., Goodenough, J.: Contextual Correlates of Synonymy. Communications of the ACM 8(10), 627–633 (1965)
24. Schuetze, H.: Automatic word sense discrimination. Computational Linguistics 24(1), 97–123 (1998)
25. Weeds, J., Weir, D., McCarthy, D.: Characterising measures of lexical distributional similarity. In: Proceedings of the 20th International Conference on Computational Linguistics. COLING '04, Association for Computational Linguistics (2004)

8 Appendix

8.1 Normalisation schemes

The raw co-occurrence counts are normalised using two measures:

- Probability Ratio

$$\frac{P(w,f)}{P(w)P(f)}$$

  where $P(w,c)$ is the probability that words w and feature f have occurred together, and $P(w)$ and $P(f)$ are probabilities of occurrences of w and f. This measure tells us how often w and f were observed together in comparison to how often they would have occurred were they independent.
- Positive Pointwise Mutual Information (PPMI)

$$\max(log(\frac{P(w,f)}{P(w)P(f)}), 0)$$

  This is the positive version of the logarithm of probability ratio, where the negative logarithmic values are sent to 0.

8.2 Formulae for computing entailment

APinc is the average precision applied to feature inclusion. It measures a ranked version of feature inclusion on vectors $\overrightarrow{u}$ and $\overrightarrow{v}$, from highest to lowest:

$$\mathit{APinc}(u,v) = \frac{\sum_r [P(r) \cdot \mathit{rel}'(f_r)]}{|F(\overrightarrow{u})|} \tag{1}$$

In the above, f_r is the feature in $\overrightarrow{u}$, denoted by $F(\overrightarrow{u})$, with rank r; $P(r)$ is the precision at rank r, which measures how many of $\overrightarrow{v}$'s features are included at rank r in the features of $\overrightarrow{u}$, and $\mathit{rel}'(f_r)$ is a relevance measure reflecting how important f_r is in $\overrightarrow{v}$. It is computed as follows:

$$rel'(f) = \begin{cases} 1 - \frac{\mathit{rank}(f, F(\overrightarrow{v}))}{|F(\overrightarrow{v})|+1} & f \in F(\overrightarrow{v}) \\ 0 & o.w. \end{cases} \tag{2}$$

BAPinc balances *APinc* with the *LIN* degree of similarity between the vectors. *BAPinc* was developed in [14] after realising that *APinc* returns poor results when the vectors had a radically different number of non-zero features; the *LIN* measure was included to balance out the extra dimensions of the longer vector.

$$\mathit{BAPinc}(u,v) = \sqrt{\mathit{LIN}(u,v) \cdot \mathit{APinc}(u,v)} \tag{3}$$

LIN is a similarity measure between vectors and was defined in [22]. It can be replaced with any other similarity measure, such as the cosine measure.

SAPinc is a measure developed in [12], based on *BAPinc*, but for dense vectors. Whereas *APinc* and *BAPinc* were developed to compute the degree of entailment between word vectors, which are usually sparse since word vectors live in high dimensional spaces (e.g. 5000), *SAPinc* was developed to deal with phrase and sentence vectors. These are obtained by composing the vectors of words in lower dimension (e.g. 300), where the compositional operators accumulate the information and return dense results.

$$\mathit{SAPinc}(u,v) = \frac{\sum_r [P(r) \cdot \mathit{rel}'(f_r)]}{|\overrightarrow{u}|} \tag{4}$$

Here, $P(r)$ and $rel'(f_r)$ are defined differently, as shown below:

$$P(r) = \frac{\left|\{f_r^{(u)} | f_r^{(u)} \leq f_r^{(v)}, 0 < r \leq |\overrightarrow{u}|\}\right|}{r} \tag{5}$$

$$rel'(f_r) = \begin{cases} 1 & f_r^{(u)} \leq f_r^{(v)} \\ 0 & o.w. \end{cases} \tag{6}$$

For more explanations on these measures please see [12, 13].

8.3 Experimental results for a second sample

The results of the experiment of Section 6, with PPMI and probability ratio matrices on the second 1000 sample of the dataset are presented in Figure 7.

Fig. 7. Results of the non-commutative conjunction experiment with the PPMI and probability ratio on the 2nd sample of dataset.

	inclusion	APinc	BAPinc	SAPinc	SBAPinc
PPMI non-comm.	0.58	0.64	0.60	0.81	0.80
PPMI: comm.	0.57	0.61	0.58	0.79	0.78
Prob: non-comm.	0.57	0.63	0.60	0.82	0.80
Prob: comm.	0.56	0.60	0.58	0.80	0.78

Similar to the results presented in the paper, the non-commutative operation perfoms better on recognising the non-commutative conjunctive entailments.