The Elements for Logic
In Compositional Distributional Models of Meaning

Joshua Steves: 1005061

St Peter's College

University of Oxford

A thesis submitted for the degree of

MSc Mathematics and Foundations of Computer Science

2016

Acknowledgements

I would like to acknowledge my supervisor Bob Coecke, as well as the other members of the Concepts group, Martha Lewis, Dan Marsden, Maaike Zwart, and Fabrizio Genovese, whose thoughts, insights, and guidance helped lead me to the results appearing in this thesis.

Abstract

Distributional compositional categorical models of meaning provide a formalism of natural language by combining distributional vector based models of word meaning and compositional models of grammar using the shared structure of a compact closed category. Word meanings and grammatical type-checking reductions are combined in the category of finite-dimensional Hilbert spaces, inspired by the mathematical formalism of quantum mechanics and quantum information protocols. The action of logic in these compositional distributional models is not immediately clear because logical functional words do not have a meaning in the distributional sense. The main elements of logic are entailment, negation, conjunction, and disjunction, which we will define within the current compositional distributional models of meaning. An entailment relation can be defined in these models by extending the models to include a natural notion of lexical entailment. Selinger's CPM-construction on any compact closed category provides the necessary extension. Applied to vectors in a Hilbert space, this extension leads to Von Neumann's density matrices, widely used in modelling quantum physics. In this enhanced framework, it becomes possible to model entailment using a canonical ordering on density matrices due to Löwner. We will define negation using Bell-states and a unitary map operating on the sentence meaning space. Negation defined in this way will lead to a natural interaction with the entailment structure. Selinger's CPM construction returns a compact closed category as well, so the construction can be iterated to model multiple features of language. Applying the CPM-construction to density matrices yields dual density matrices, which linguistically will model ambiguity and lexical entailment together. After iterating the CPM-construction we can define conjunction and disjunction using environment structures to take two different kinds of sums, one correlating to ambiguity and the other correlating to entailment. Dual density matrices can be transformed into density matrices allowing comparisons in CPM^2 using the same entailment relation.

Contents

Chapter 1

Introduction

1.1 Background

Developing a mathematical model for meaning in language has been an important and challenging problem for many years. Aside from being a generally interesting problem intellectually and improving our understanding of how we use language, mathematical models for natural language have important applications in computer science. Models of meaning are relied on for natural language processing tasks such as information retrieval, automatic summarization, machine translation, and many more. Two main approaches to finding a formalization of language have been pursued in the past: distributional and symbolic.

Distributional theories of meaning use statistical techniques to extract word meanings from large corpora. The distributional hypothesis is that a word's meaning can be understood by the words which it appears together with. Distributional models, such as those used by Google's search algorithms, represent the meaning of words as vectors in finite-dimensional vector spaces by using co-occurrence statistics from corpora. Symbolic theories describe the compositional properties of grammatical structure in language. Lambek pioneered the use of Pregroups to model syntax and grammar of natural language [17]. Pregroup models of grammar effectively type-check sentences determining when a string of words is a grammatically correct sentence. However, these symbolic theories have no way of determining individual word meanings. Distributional, vector based models on the other hand extract word meanings, but have no obvious compositional structure.

Using ideas from the mathematical formalism of quantum mechanics, Coecke et al. in [10], developed a Distributional Compositional Categorical (DisCoCat) model of meaning bridging the gap between the two previously incompatible approaches. The authors were able to make this passage by recognizing that Finite Dimensional Vector

Spaces and Pregroups share the categorical structure of a compact-closed category. These new models were able to outperform previous linguistic models because they provided a means to compare meaning at the sentence and word level [13, 15].

An open problem in the DisCoCat models is how language and logic interact. One issue in modelling logic within the current framework is that functional words, such as *not, and, or,* etc., do not have a meaning in the distributional sense. Progress has been made in modelling certain kinds of functional words such as relative pronouns by using the structure of Frobenius algebras [31, 32].

In [23], Von Neumann introduced density operators to describe systems in quantum mechanics that could be in a variety of states, each with a given probability. Density operators are not standard probability distributions and carry important extra structure. We can obtain density matrices from our vector based models by following a categorical construction due to Selinger, called the CPM-construction [34]. Density operators can be applied to computational linguistics to cover two different properties of natural language: ambiguity and lexical entailment [4, 26]. The CPM-construction can also be iterated in order to model ambiguity and entailment together using dual density matrices.

Towards understanding logic in the DisCoCat models, [4] described an entailment relation based on a partial ordering of density matrices. Negation, conjunction, and disjunction remain to be defined in DisCoCat models. In this paper I define the key elements to understand logic in the DisCoCat models: entailment, negation, conjunction, and disjunction.

1.2 Outline and New Contributions

The goal of this paper is to define and describe the necessary elements to understand and formalize logic in natural language. The necessary components needed for logic are models of entailment, negation, conjunction, and disjunction. I will describe in this paper how to model these four elements needed for logic and describe how they interact. This paper is divided into three main chapters organized by the level of category being used and discussed ($\mathcal{C}$ vs. CPM($\mathcal{C}$) vs. CPM^2($\mathcal{C}$)).

Chapter 2 provides an overview of the DisCoCat model and discusses the relevant mathematical formalism necessary to understand these models of meaning. I begin by providing the necessary elements of category theory and describe a graphical calculus for compact closed categories. I separately discuss the compositional structure of Pregroups and the distributional vector based models and show how these can be combined through the mathematics of compact closed categories. I further show how meanings of words and sentences can be compared in the DisCoCat model using the inner product of the vector spaces. Finally I define a Frobenius algebra over our categories, which can be used to model relative pronouns, but will also be helpful in the constructions of chapter 4.

I will discuss in Chapter 3 the CPM-construction and how density matrices can be used to model entailment and ambiguity. I will describe a partial order structure on density matrices that allows for a logical description of entailment that operates at both the word and sentence level as in [4]. The first new contributions to the existing models appears at the end of chapter 3; I define a map to model negation and analyze how this model of negation interacts with the entailment structure.

Chapter 4 begins with a description of environment structures, which are used in [6] to give an axiomatic description of Selinger's CPM-construction. The CPM-construction can also be iterated (CPM^2) to move from density matrices to dual-density matrices, similar to the way we passed from vectors to density matrices. Iterating the CPM construction allows us to model both ambiguity and lexical entailment together. I contribute a model of conjunction and disjunction (*and* and *or*) using the extra features that dual density matrices provide over normal density operators. Finally, I will analyze how conjunction and disjunction interact with the entailment relation and negation as defined in chapter 3.

Chapter 2

Distributional Compositional Categorical Models of Meaning

Distributional Compositional Categorical models of meaning, as the name suggests, unite compositional and distributional theories of linguistics via category theory. We begin by introducing the necessary category theory, that of compact closed categories. We will present briefly compositional theory using Lambek's Pregroups and show how Pregroups have the structure of a compact closed category. Additionally we will describe a vector based distributional model of meaning, revealing again how it is realized in a compact closed category. Because Pregroups and vector spaces share categorical structure we can combine them using a strongly monoidal functor. Finally, we will introduce Frobenius algebras as extra structures we can add to our categories that allow us to enrich the linguistic models.

2.1 Categorical Framework

2.1.1 Monoidal Categories

Definition 2.1.1. (Monoidal Category)

A monoidal category $(\mathcal{C}, \otimes, I, \alpha, \lambda, \rho)$ consists of:

- A family of objects $Ob(\mathcal{C})$;

- A set of morphisms $\mathcal{C}(A, B)$ for every ordered pair of objects;

- A sequential composition operation $\circ$, such that for an ordered triple of objects (A, B, C) each $f : A \to B$ and $g : B \to C$ has a composite $g \circ f : A \to C$, and this composition is associative: $(h \circ g) \circ f = h \circ (g \circ f)$;

- A tensor functor $\otimes$ acting on objects by $(A, B) \mapsto A \otimes B$ and on morphisms by $(f : A \to B, g : C \to D) \mapsto f \otimes g : A \otimes C \to B \otimes D$ that is bifunctorial:

$$(f \otimes g) \circ (h \otimes k) = (f \circ h) \otimes (g \circ k)$$

- A unit object $I \in Ob(\mathcal{C})$;

- A natural isomorphism called an associator α satisfying for every triple of object (A, B, C),

$$\alpha_{A,B,C} : A \otimes (B \otimes C) \cong (A \otimes B) \otimes C$$

  and natural isomorphisms called left and right unitors satisfying for each object

$$\lambda_A : I \otimes A \cong A \qquad \rho_A : A \otimes I \cong A$$

  and

$$\lambda_I = \rho_I : I \otimes I \cong I$$

The structural isomorphisms λ, ρ, and α are subject as well to certain coherence conditions known as the triangle and pentagon equations. If all of these structural isomorphisms are identities then the monoidal category is said to be *strict*. We will assume from this point on that the categories in which we are working are strict, relying on the coherence theorems of MacLane, in particular using the fact that every monoidal category is categorically equivalent to a strict monoidal category [21].

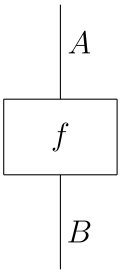
Some examples of monoidal categories are:

- **(F)Hilb**: The category of (finite-dimensional) Hilbert spaces and bounded linear maps with the tensor product being the regular tensor product of Hilbert spaces

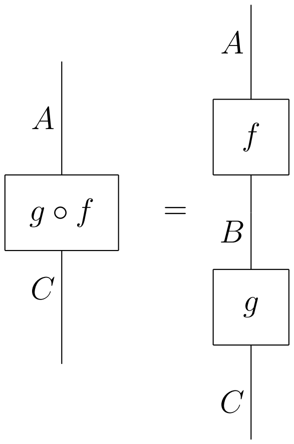
- **Set**: The category of sets and functions with tensor product given by the Cartesian product

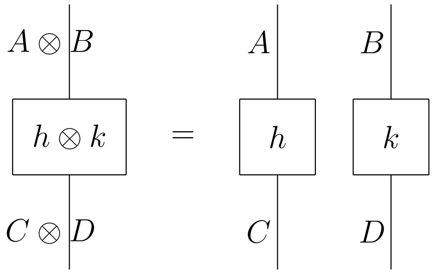
- **Rel**: The category of sets and relations with tensor product given by the Cartesian product

Monoidal categories allow a nice representation in graphical form. We can represent objects in our category as wires and morphisms as boxes with input and output wires. Interpreting our diagrams from top to bottom we can depict for example a morphism $f : A \to B$ as:

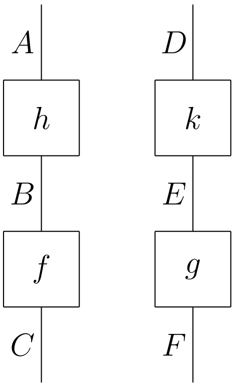


Identity maps will be represented by straight wires, and the monoidal unit will be drawn as an empty wire. Sequential composition $\circ$ and parallel composition $\otimes$ are represented in this graphical language by vertical and horizontal composition of diagrams:

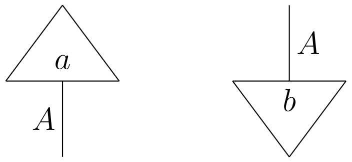




One main feature of this graphical representation is that many of the underlying structural equations in our categories become trivial. For example consider the bifunctoriality equation, also called the interchange law: $(f \otimes g) \circ (h \otimes k) = (f \circ h) \otimes (g \circ k)$. Both sides of this equation are represented identically in the graphical language:

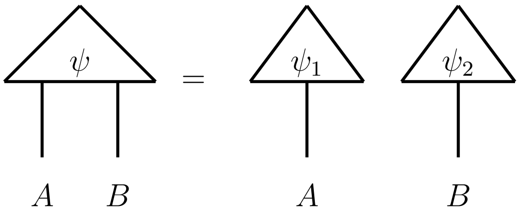


Since we do not draw the monoidal unit I, we can represent states and effects (maps of the form: $a : I \to A$ and $b : A \to I$) as special boxes which we draw as triangles:



We can combine together these basic building blocks to construct more complex diagrams. Two diagrams are equivalent if they are equivalent up to planar isotopy. Note however, that we are not yet allowed to cross wires.

Definition 2.1.2. (Entangled States) A joint state on a composite system is *separable* if it is the tensor product of two distinct states.



A joint state is *entangled* if it is not separable.

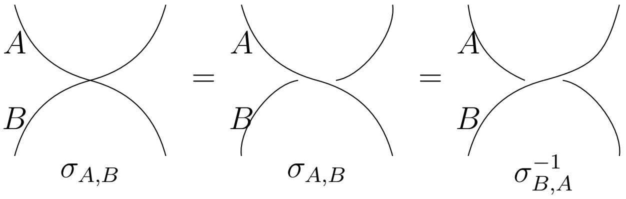
Entanglement is a complex phenomenon in physics that fundamentally distinguishes classical and quantum mechanics. In models of classical physics there is no notion of entanglement, or in other words every joint state is separable. The reason for this is that classical physics lives in **Set** where the Cartesian product is the tensor product and a categorical product. In quantum mechanics, which is modelled in **Hilb**, this is not true. We will see later that entangled states are necessary to the flow of information and meaning in natural language, which we will model in **FHilb**.

2.1.2 Compact Closed Categories

Definition 2.1.3. (Symmetric Monoidal Category)

A symmetric monoidal category is a monoidal category equipped with a swap map, or symmetry, for every pair of objects $\sigma_{A,B} : A \otimes B \cong B \otimes A$ such that $\sigma_{B,A} \circ \sigma_{A,B} = id_{A\otimes B}$.

We can extend our graphical calculus for monoidal categories by adding a braiding represented by crossing wires. In a symmetric monoidal category we have:



Definition 2.1.4. (Compact Closed Category)

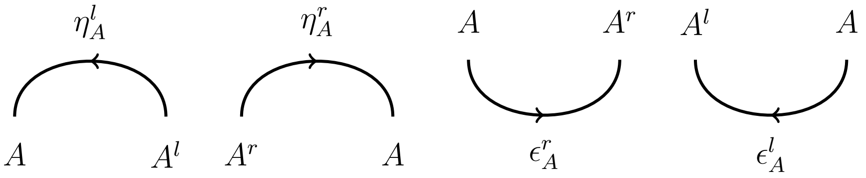
A compact closed category is a symmetric monoidal category where every object $A \in Ob(\mathcal{C})$ has a corresponding left and right dual object $A^l, A^r \in Ob(\mathcal{C})$ and related structural isomorphisms $\eta^r_A : I \cong A^r \otimes A$, $\eta^l_A : I \cong A \otimes A^l$, $\epsilon^r_A : A \otimes A^r \cong I$, and $\epsilon^l_A : A^l \otimes A \cong I$. These structural isomorphisms satisfy the snake equations:

$$(1_A \otimes \epsilon^l_A) \circ (\eta^l_A \otimes 1_A) = 1_A = (\epsilon^r_A \otimes 1_A) \circ (1_A \otimes \eta^r_A)$$

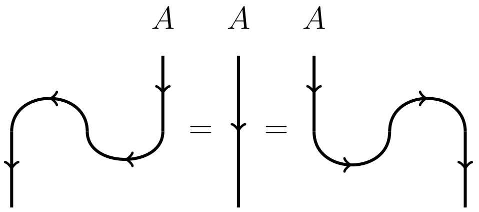
$$(\epsilon^l \otimes 1_{A^l}) \circ (1_{A^l} \otimes \eta^l_A) = 1_{A^l}$$

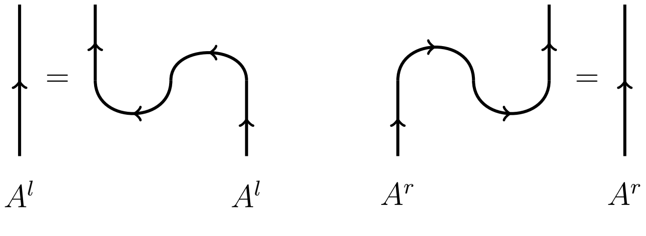
$$(1_{A^r} \otimes \epsilon^r_A) \circ (\eta^r_A \otimes 1_{A^r}) = 1_{A^r}$$

We can draw the structural isomorphism η and ϵ as cups and caps in the graphical language. Additionally it will be helpful to orient our diagrams to avoid confusion. There are four types of cups and caps:

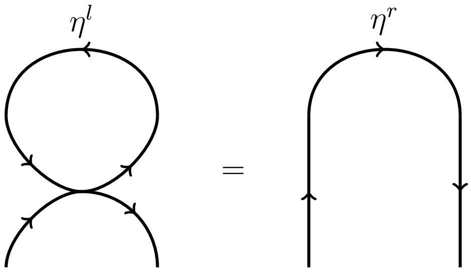


And the snake equations look like yanking straight a snake in a wire:





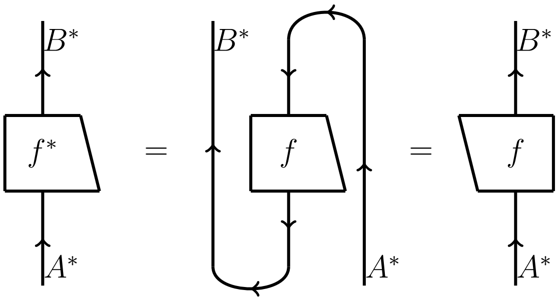
Because the category is symmetric, there is in fact a unique dual A^* for every object A that is both its left and right dual object. The orientation of the arrows on the object wires indicates whether the wire is referring to the object or its dual, and the left and right cups and caps are related in the following way using the symmetry:



FHilb is an example of a compact closed category. In **FHilb** every object V is its own left and right dual. Say $\{|i\rangle\}_i$ is a basis for V then the cups and caps are given by the maps $\eta : \mathbb{R} \to V \otimes V :: 1 \mapsto \sum_i |i\rangle \otimes |i\rangle$ and $\epsilon : V \otimes V \to \mathbb{R} :: |v\rangle \otimes |w\rangle \mapsto \langle v|w\rangle$

We can also compose our morphisms with cups and caps to get the notion of a dual morphism, also called a linear operator transpose because of its relation to the matrix transpose in linear algebra.

Definition 2.1.5. (Dual Morphism) Given a morphism $f : A \to B$ its dual is the morphism $f^* : B^* \to A^*$ defined graphically:

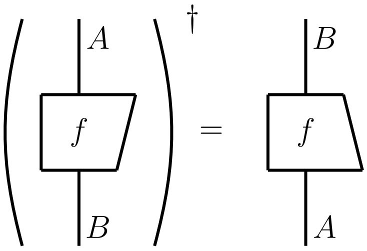


Definition 2.1.6. (Dagger Functor)

A dagger functor on a category, $\mathcal{C}$ is a functor $\dagger : \mathcal{C} \to \mathcal{C}$, that is contravariant ($f : A \to B \mapsto f^\dagger : B \to A$), identity on objects ($A^\dagger = A$), and involutive ($(f^\dagger)^\dagger = f$).

A Dagger Category is a category equipped with a dagger functor.

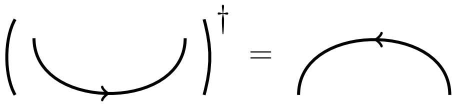
Graphically we can represent daggers by a vertical flip:



The dagger functor on a symmetric monoidal dagger category respects the monoidal structure and the symmetry:

$$(g \circ f)^\dagger = f^\dagger \circ g^\dagger \quad (f \otimes g)^\dagger = f^\dagger \otimes g^\dagger \quad \sigma^\dagger_{A,B} = \sigma_{B,A}$$

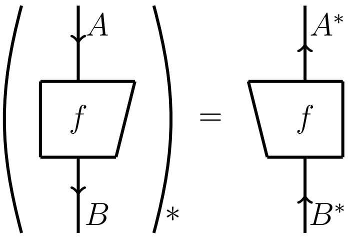
In a dagger compact closed category we also have $\epsilon^\dagger_A = \eta_{A^*}$:



The dagger of a map in **FHilb** is given by its adjoint.

Combining the dagger and transpose we can obtain the notion of a conjugate.

Definition 2.1.7. Given a morphism $f : A \to B$ its conjugate is a map $f_* = (f^\dagger)^*$.



2.2 Compositional Meaning through Pregroups

In language words have grammatical types that combine in specific ways to form sentences according to the grammar of the language. For example a simple transitive sentence such as, ”Mathematicians study math,” consists of a subject noun, a transitive verb and an object noun, which together combine to form the sentence. Lambek formalized these notions of grammar in natural language using Pregroups [17].

Definition 2.2.1. (Partially Ordered Monoid)

A Partially ordered monoid (P, $\leq$, $\cdot$, 1) is a partially ordered set together with a monoid multiplication -·- and unit 1, such that if $p, q, r \in P$ and $p \leq q$ the following equalities hold: $r \cdot p \leq r \cdot q$ and $p \cdot r \leq q \cdot r$.

Definition 2.2.2. (Pregroup Algebra)

A pregroup algebra (P, $\leq$, $\cdot$, 1, (-)^r, (-)^l) is a partially ordered monoid for which every element $p \in P$ has left and right adjoints p^l and p^r satisfying: $p^l \cdot p \leq 1 \leq p \cdot p^l$ and $p \cdot p^r \leq 1 \leq p^r \cdot p$.

We can think of $\leq$ as a type reduction → as in the Lambek Calculus.

Definition 2.2.3. (Pregroup Grammar)

A Pregroup Grammar $\mathcal{G}$ is a pregroup algebra freely generated over a set of basic types $\mathcal{B}$ with a designated end type and a type dictionary that assigns elements of a pregroup to the vocabulary of a language.

For our linguistic models we will assign $\mathcal{B} = \{n, s\}$ where n represents a noun type and s a well-formed sentence. We can interpret other types of words by combining these atomic types. For example a transitive verb will have type: $n^r \cdot s \cdot n^l$. The grammatical pregroup reduction for a transitive sentence will then look like:

$$n \cdot (n^r \cdot s \cdot n^l) \cdot n = (n \cdot n^r) \cdot s \cdot (n^l \cdot n) \leq 1 \cdot s \cdot 1 \leq s$$

.

A sentence is well typed if it reduces in this way to the atomic type s. We can also consider noun phrases as grammatical if they reduce to the type n. For example the noun phrase, "People who study math," has the pregroup reduction:

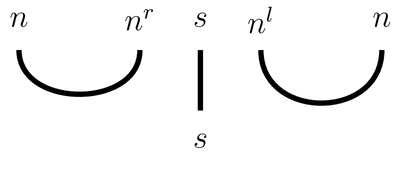
$$n \cdot (n^r \cdot n \cdot s^l \cdot n) \cdot (n^r \cdot s \cdot n^l) \cdot n \leq n$$

2.2.1 Pregroups as Compact Categories

Definition 2.2.4. Preg$_{\mathcal{G}}$ is the compact category whose objects are the elements of a pregroup grammar $\mathcal{G}$ and every pair of objects has either one or no morphisms between them. A pair p, q has a morphism $p \to q$ if $p \leq q$. The monoidal tensor is the monoidal multiplication $\cdot$ of the pregroup. The compact closure maps are:

$$\eta^l = (1 \leq p \cdot p^l) \quad \eta^r = (1 \leq p^r \cdot p) \quad \epsilon^l = (p^l \cdot p \leq 1) \quad \epsilon^r = (p \cdot p^r \leq 1)$$

It is easy to check that **Preg** satisfies the axioms for a compact closed category. The map for a pregroup reduction for a positive transitive sentence is given diagrammatically by:



2.3 Distributional Meaning

The goal in distributional semantics is to represent the meaning of a word as a vector in a finite dimensional vector space. We will work in the category of **FHilb**, which has the extra structure of an inner product space. A meaning vector for a word is collected from a large corpus of text by choosing some suitable basis and counting how many times a word appears near the basis words and then normalizing the vectors. For example if your basis is the set of words machine, person, animal, sea, land, air, you might obtain a vector $[1/\sqrt{2}, 0, 0, 1/\sqrt{2}, 0, 0] = \frac{1}{\sqrt{2}}|\text{machine}\rangle + \frac{1}{\sqrt{2}}|\text{sea}\rangle$ corresponding to the word *ship*, which appeared half the time in the same sentence as machine and half the time in the same sentence as sea, but never in the same sentence as person, animal, land, or air. Meanings of words can then be compared using the inner product to determine the angle between two words. For example, let $|\text{sailor}\rangle = \frac{1}{\sqrt{2}}|\text{person}\rangle + \frac{1}{\sqrt{2}}|\text{sea}\rangle$ and $|\text{pilot}\rangle = \frac{1}{\sqrt{2}}|\text{person}\rangle + \frac{1}{\sqrt{2}}|\text{air}\rangle$, then we can compare the meanings of pilot, sailor, and ship:

$$\langle\text{pilot}|\text{sailor}\rangle = \frac{1}{2} \quad \langle\text{ship}|\text{sailor}\rangle = \frac{1}{2} \quad \langle\text{pilot}|\text{ship}\rangle = 0$$

2.4 Combining Syntax and Semantics

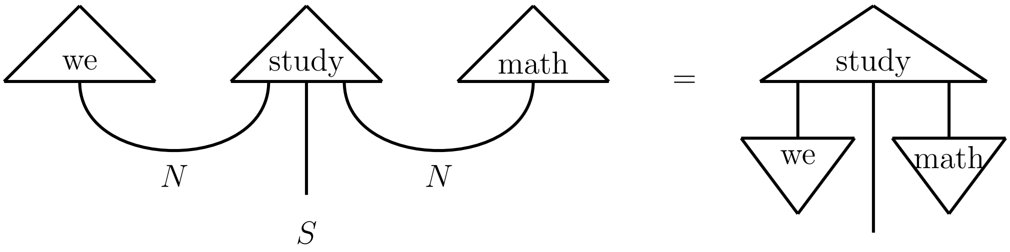
We have shown how to model compositional structure of grammar in the category **Preg** and distributional semantics in the category **FHilb**, which are both compact closed categories. These two models are combined via a strongly monoidal functor which maps **Preg** into **FHilb** as in [31].

Let F : **Preg** $\to$ **Fhilb** be the monoidal functor that assigns atomic types to vector spaces: $F(1) = I$, $F(n) = N$ and $F(s) = S$. Pregroup reductions are sent to linear maps, and the cups and caps in **Preg** are sent to the cups and caps **FHilb**. Strongly monoidal functors preserve the compact structure meaning $F(n^l) = F(n)^*$, and by monoidality complex types are mapped to tensor products of vector spaces, for example $F(n^r \cdot s \cdot n^l) = F(n^r) \otimes F(s) \otimes F(n^l) = N \otimes S \otimes N$.

Definition 2.4.1. Let w_i be a word with meaning vector $|w_i\rangle$ and α be the pregroup reduction map for a sentence. The meaning of a sentence $w_1 w_2 ... w_n$ is given by $|w_1 ... w_n\rangle = F(\alpha)(|w_1\rangle ... |w_n\rangle)$

For example consider the sentence, ”We study math,” which has pregroup reduction map $\epsilon_n \otimes id_s \otimes \epsilon_n$. The sentence meaning vector is given by

$$F(\epsilon_n \otimes id_s \otimes \epsilon_n)(|\text{We}\rangle \otimes |\text{study}\rangle \otimes |\text{math}\rangle) = (\epsilon_n \otimes id_s \otimes \epsilon_n)(|\text{We}\rangle \otimes |\text{study}\rangle \otimes |\text{math}\rangle)$$

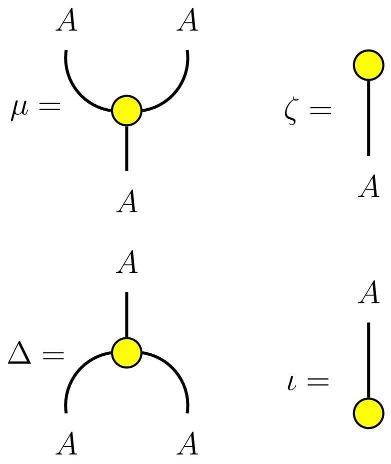


The result is a vector in the sentence space S. These sentence vectors can be compared just as the word vectors were before by taking the inner product in the sentence space.

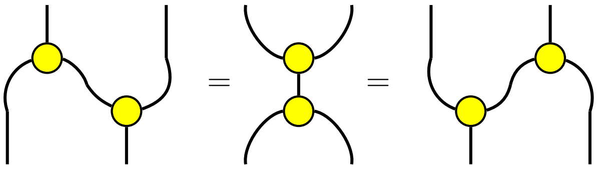
2.5 Frobenius Algebras

Frobenius Algebras are a type of extra structure we add on top of the categories we have already discussed. In physics and mathematics Frobenius algebras capture the intuition of copying bits of information or the inverse of copying, which acts like matching. In linguistics, Frobenius Algebras have been used most successfully to model relative pronouns such as whose, which, that [31, 32]. They can also be used as in [26] to model some types of adjectives. In [14], a model for coordination and conjunction is proposed using Frobenius algebras. In our categories copying individual outputs of an entangled state is not the same as copying the whole state, which causes some issues when we try to use these maps to model aspects of language such as conjunction. In particular, Frobenius models of conjunction do not provide a clear way to define or relate to disjunction. I will propose a different model of "and" and "or" in chapter 4 that captures the entailment relation between the two words in natural language which we use similarly to conjunction and disjunction in logical contexts. However, we will still need Frobenius algebras for the constructions in chapter 4.

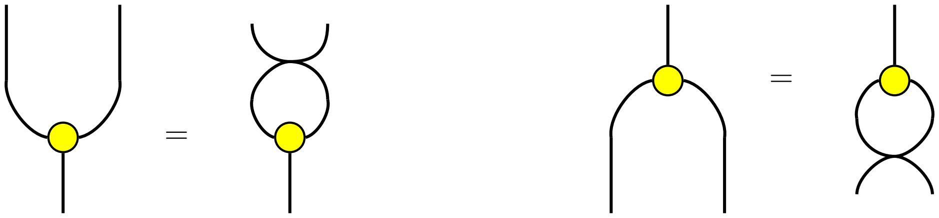
Definition 2.5.1. A Frobenius Algebra over a monoidal category is a tuple $(A, \Delta, \iota, \mu, \zeta)$ representing for an object A, an internal monoid (A, μ, ζ) and an internal comonoid (A, Δ, ι):



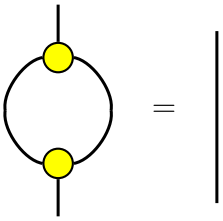
satisfying the Frobenius condition:



Definition 2.5.2. A *commutative Frobenius Algebra* over a braided monoidal category is a Frobenius Algebra $(A, \Delta, \iota, \mu, \zeta)$ whose monoid and co-monoid multiplication, μ and Δ, are commutative and co-commutative respectively:

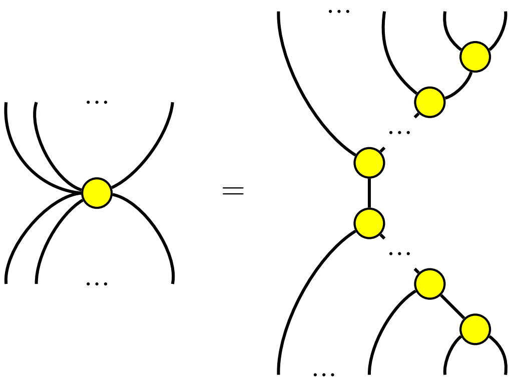


A †-Frobenius Algebra over a †-monoidal category satisfies $\Delta = (\mu)^\dagger$ and $\iota = (\zeta)^\dagger$. A *special* Frobenius Algebra satisfies:

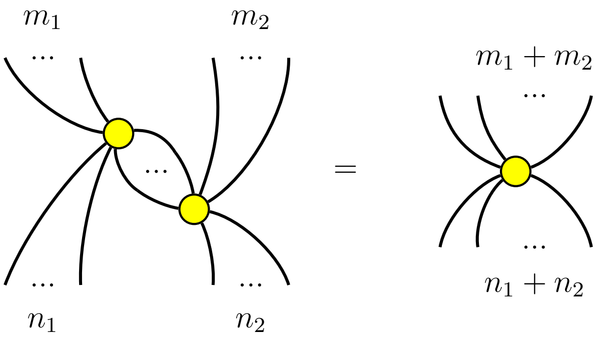


Using the associativity, unitality, co-associativity, and co-unitality of monoids and comonoids together with the Frobenius condition, we can rearrange any connected morphism built from $\Delta, \iota, \mu, \zeta$, and id with $\circ$ and $\otimes$ into a normal form. A detailed explanation of this processes can be found in [22]. These types of maps in their normal form are given a special name.

Definition 2.5.3. (Spiders) Given a category with a Frobenius algebra a spider is map $A^{\otimes m} \to A^{\otimes n}$ of the following form:

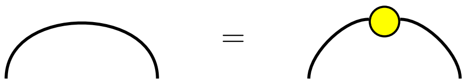


Theorem 2.5.1. (Spider Fusion) Spiders from a special commutative †-Frobenius Algebra compose by fusing together:



Proof. The proof follows directly from the ability to write spiders in a normal form and from the axioms of a special commutative †- Frobenius Algebra. □

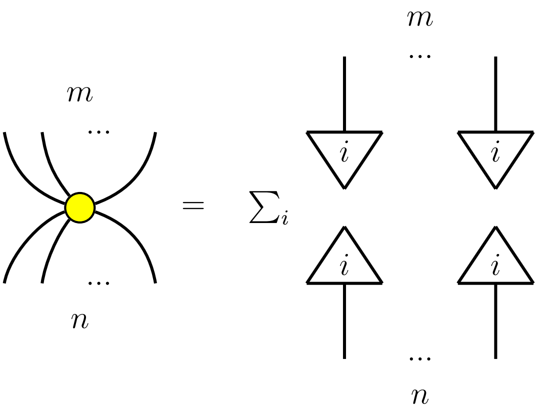
We have actually already encountered some special spiders: the compact closure maps of our compact closed categories



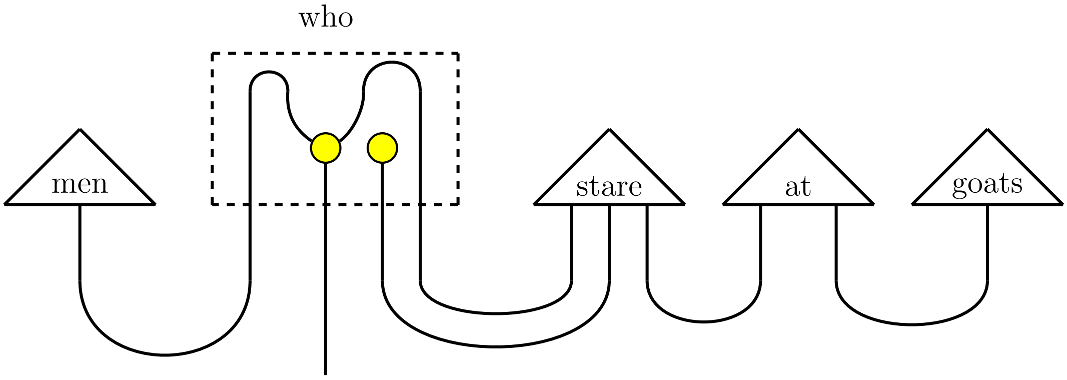
Our linguistic models live in the category **FHilb**. We can define a Frobenius Algebra on **FHilb** in terms of operations on basis elements. Let $V \in Ob(\mathbf{FHilb})$ be a finite-dimensional Hilbert space with an orthonormal basis $\{|i\rangle\}_i$. We define the Frobenius units and multiplications by their actions on the basis elements:

$$\Delta : |i\rangle \mapsto |i\rangle \otimes |i\rangle \quad \iota : |i\rangle \mapsto 1 \quad \mu : |i\rangle \otimes |j\rangle \mapsto \delta_{ij}|i\rangle \quad \zeta : 1 \mapsto \sum_i |i\rangle$$

It is easy to check that these do satisfy the axioms of a special commutative † Frobenius algebra. Spiders in **FHilb** become:



The main use of Frobenius algebras in categorical models of meaning is to model relative pronouns. For example the noun phrase, ”men who stare at goats,” takes a transitive sentence, ”men stare at goats” and turns it into a noun by matching the dimensions of the vector for *men* that occur as subjects of *stare at goats*.



Chapter 3

Density Matrices: Modelling Entailment, Ambiguity, and Negation

Von Neumann introduced density operators to model uncertainty about the state of a system in quantum mechanics [23]. Recall from linear algebra that positive operators are self-adjoint, positive matrices. By positive, we mean that for a linear map ρ in a Hilbert space V, ρ is positive if $\forall v \in V$, $0 \leq \langle v|\rho|v\rangle$. A density matrix is a positive operator with trace 1. Positive operators and density matrices are subsumed in the categorical language through the framework of completely positive maps. Selinger formalized the categorical construction of completely positive maps called the CPM-construction [34], which can be applied to any compact category.

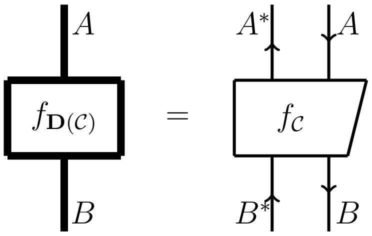
Lexical entailment and ambiguity are two important features of language. These features manifest in what are called hyponyms/hypernyms and homonyms. A word is a hyponym if it is a more specific case of a general word. Salmon is a hyponym of fish, and conversely fish is a hypernym of salmon. Homonyms are words that have the same spelling but different meanings, such as bank, which means a financial institution or the side of a river. In [4, 26], the authors show how to use density matrices and positive maps to encode lexical entailment and ambiguity, respectively.

3.1 Doubling and the CPM Construction

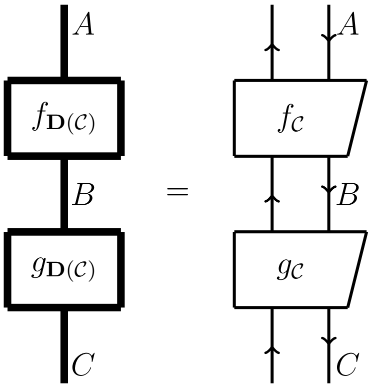
Definition 3.1.1. (Doubling Construction) Given a dagger compact closed category $\mathcal{C}$ the doubled category $\mathbf{D}(\mathcal{C})$ is defined as follows:

- The objects of $\mathbf{D}(\mathcal{C})$ are the same as the objects of $\mathcal{C}$.

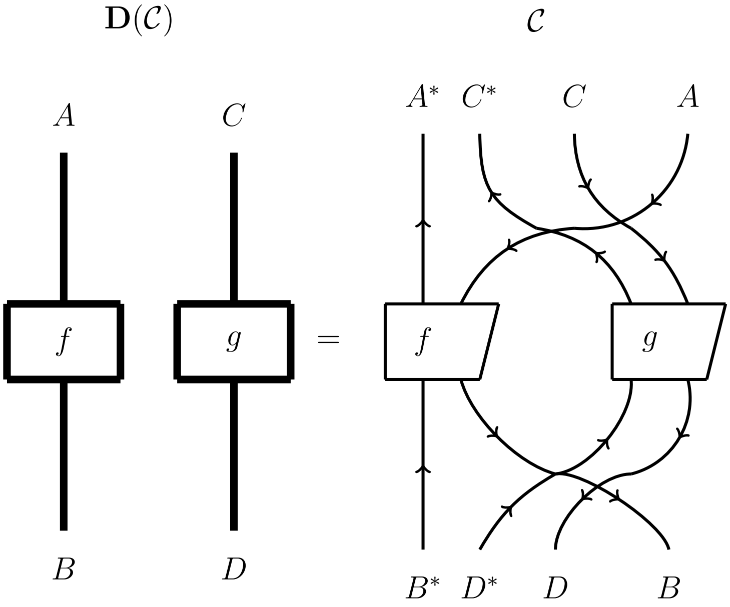
- The morphisms in $\mathbf{D}(\mathcal{C})(A, B)$ are morphisms in $\mathcal{C}(A^* \otimes A, B^* \otimes B)$.



- Sequential composition follows exactly from sequential composition of the morphisms in $\mathcal{C}$.

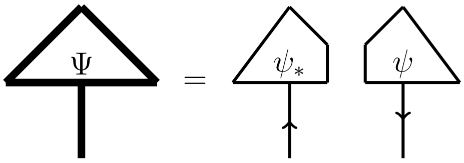


- Parallel composition via the monoidal tensor $\otimes_{\mathbf{D}}$ acts on objects by $A \otimes_{\mathbf{D}} B = A \otimes B$ and on morphisms $f : A^* \otimes A \to B^* \otimes B$, $g : C^* \otimes C \to D^* \otimes D$ by $f \otimes_{\mathbf{D}} g : A^* \otimes C^* \otimes C \otimes A \xrightarrow{\cong} A^* \otimes A \otimes C^* \otimes C \xrightarrow{f \otimes g} B^* \otimes B \otimes D^* \otimes D \xrightarrow{\cong} B^* \otimes D^* \otimes D \otimes B$.



The doubling construction gets its name because of the way we can construct maps in $\mathbf{D}(\mathcal{C})$ by "doubling" maps in $\mathcal{C}$. That is if $f : A \to B$ is a morphism of $\mathcal{C}$, then $f_* \otimes f : A^* \otimes A \to B^* \otimes B$ is a morphism in $\mathbf{D}(\mathcal{C})$. States constructed in this way have a special name:

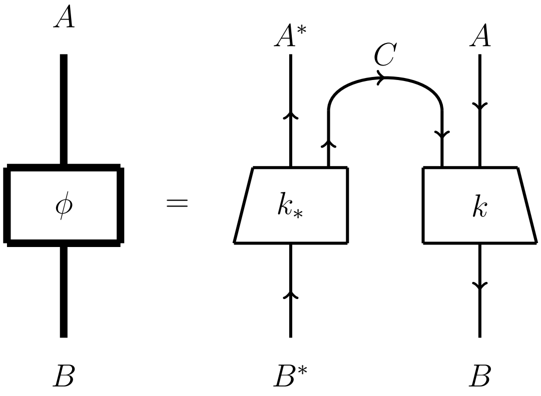
Definition 3.1.2. A state $\Psi \in \mathbf{D}(\mathcal{C})(I \otimes I, A^* \otimes A)$ is pure if it is of the form: $\psi_* \otimes \psi$ for some $\psi \in \mathcal{C}(I, A)$.



A state is called a mixed state if it is not pure.

We can directly construct mixed states by taking weighted sums of pure states. Pure states are a special example of completely positive maps.

Definition 3.1.3. (Complete Positivity) A morphism $\phi \in \mathbf{D}(\mathcal{C})(A, B)$ is completely positive if for some $k \in \mathcal{C}(C \otimes A, B)$:



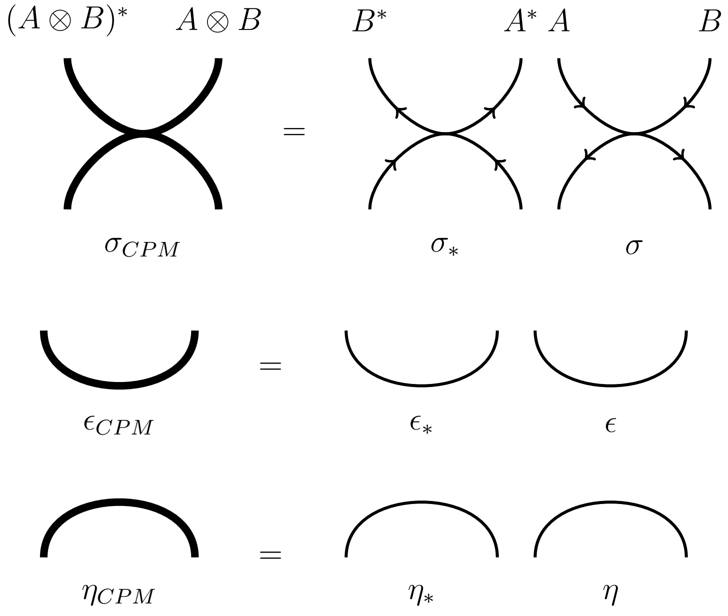
In the above definition we call the object C the ancillary system. Its clear from the graphical representation that a completely positive state is pure if the ancillary system is I. Additionally we can construct mixed states that are completely positive if we take probability distributions over pure states.

Definition 3.1.4. (CPM($\mathcal{C}$))

Let $\mathcal{C}$ be a dagger compact closed category. Then, CPM($\mathcal{C}$) is the subcategory of $\mathbf{D}(\mathcal{C})$ whose objects are the same as $\mathbf{D}(\mathcal{C})$ and whose morphisms are the completely positive morphisms of $\mathbf{D}(\mathcal{C})$.

Proposition 3.1.1. If $\mathcal{C}$ is a dagger compact closed category, then CPM($\mathcal{C}$) is a dagger compact closed category.

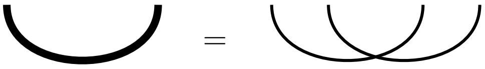
Proof. CPM($\mathcal{C}$) is a monoidal category with tensor functor given as defined previously in the doubled category $\otimes_{\text{CPM}} = \otimes_{\mathbf{D}}$. CPM($\mathcal{C}$) is a symmetric monoidal category with braiding $\sigma^{CPM}_{(A,B)} : B^* \otimes A^* \otimes A \otimes B \to A^* \otimes B^* \otimes B \otimes A = \sigma^{\mathcal{C}}_{(B^*,A^*)} \otimes \sigma^{\mathcal{C}}_{(A,B)}$. CPM($\mathcal{C}$) similarly inherits a compact structure from $\mathcal{C}$ by $\epsilon^{CPM} = \epsilon^{\mathcal{C}}_{*} \otimes \epsilon^{\mathcal{C}}$ and $\eta^{CPM} = \eta^{\mathcal{C}}_{*} \otimes \eta^{\mathcal{C}}$



The symmetry and snake equations in CPM($\mathcal{C}$) follow easily from their related equations in $\mathcal{C}$ and the definitions of the tensor product in CPM($\mathcal{C}$). The dagger functor † in CPM($\mathcal{C}$) is also inherited from the dagger on $\mathcal{C}$ and acts in the obvious way.

□

Note that because of the swaps induced in the tensor products of states and maps in CPM($\mathcal{C}$), it will be convenient to write the compact closure maps instead in the following equivalent manner:



The fact that the CPM construction returns again a dagger compact closed category is crucial to the constructions in the next chapter where we will exploit the ability to iterate the CPM construction to accommodate more features of natural language in the same model.

3.2 Ambiguity

Since **FHilb** is a compact closed category we can apply the CPM construction to encode our linguistic models in CPM(**FHilb**) instead of **FHilb**. All our states, which were vectors before now become density matrices, and we retain the same expressive power as before. The additional bonus of encoding states in this manner lies in the ability to take probabilistic mixtures of pure states. One way we can use mixed states is to encode ambiguity as in [26, 25]. Homonyms are words that have distinct meanings. However, until we hear a homonym in a sentence it is ambiguous which meaning of the word is being referred to.

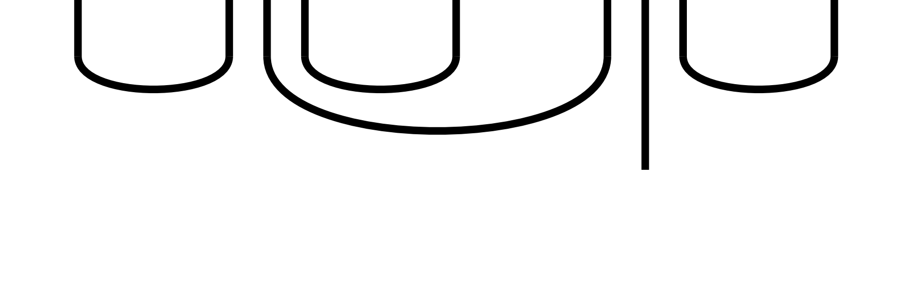
Explicitly in braket notation, given an ambiguous W with unambiguous meanings w_i:

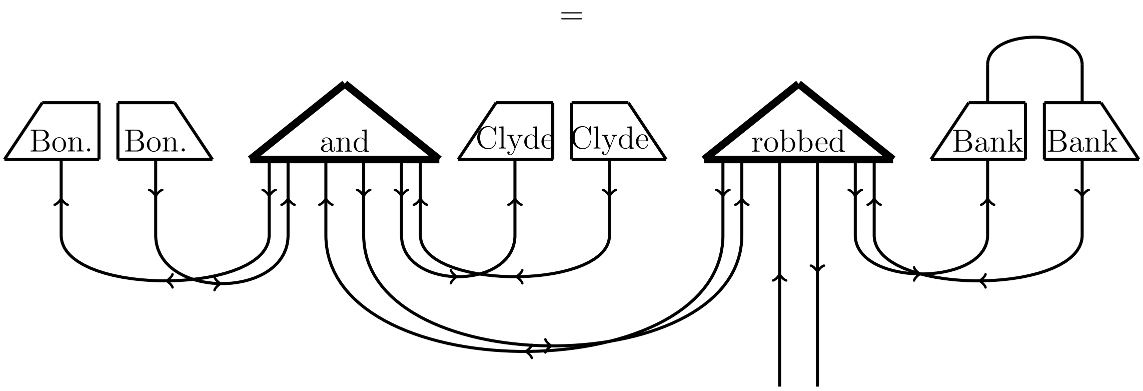
$$[W] = \sum_i p_i |w_i\rangle\langle w_i| \qquad (\sum_i p_i = 1, p_i \in [0,1])$$

For example we can represent the ambiguous word bank by a sum over its unambiguous meanings: river-bank or financial-bank.

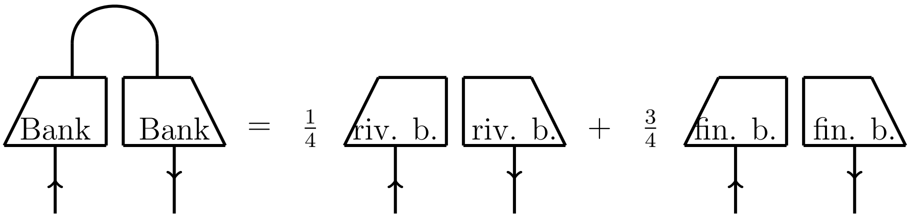
$$[bank] = \frac{1}{4}|\text{river-bank}\rangle\langle\text{river-bank}| + \frac{3}{4}|\text{financial-bank}\rangle\langle\text{financial-bank}|$$

The coefficients in the sum represent how often the unambiguous meanings of the word appeared in the corpora used to build the distributional model. Intuitively these coefficients are how likely someone is to mean one meaning of the word versus another when the word is heard outside of a sufficient context to disambiguate it. As in Quantum Mechanics, mixed states in this sense are representing some lack of knowledge about a system. We can then think of hearing the word in an unambiguous context of a sentence as measuring the state and causing the ambiguous meaning to collapse onto one of the unambiguous, pure meanings it could have. For example consider the sentence, "Bonnie and Clyde robbed banks," with the following reduction:



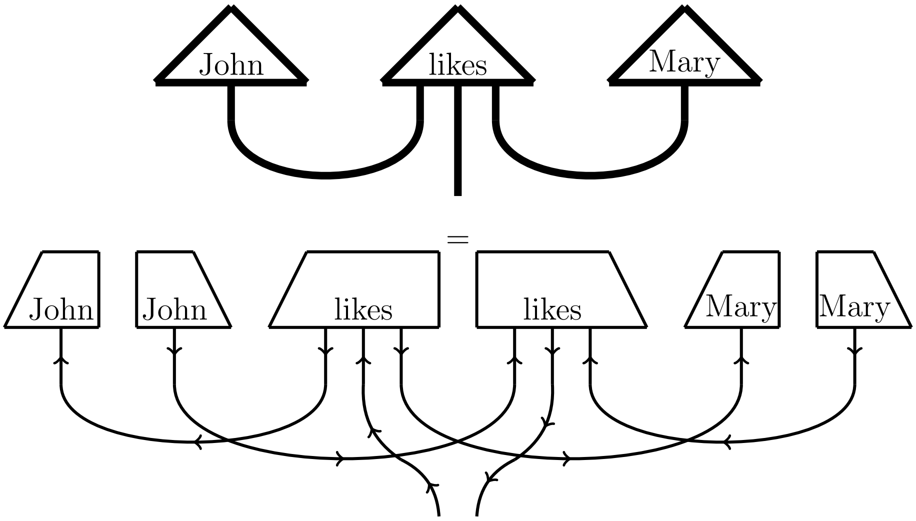


With:



The piece of the sum including river bank will evaluate to 0 in the sentence reduction and only the meaning financial bank will carry through the sentence.

If our sentence is composed of only pure states then we have the same reductions as before only doubled. For example:



3.3 Entailment and k-Hyponymy

Instead of using mixed states to model ambiguity, we could use them to model lexical entailment. Following as in [4], we represent a hypernym as a probabilistic sum over its hyponyms. Lexical entailment has a natural logical interpretation as well. A crucial feature of density matrices is that there is a partial order over them [20]. This partial order can be used as an entailment relation between words, which will allow us to make logical inferences about words and sentences. If a sentence is true for all fish then it should be true for salmon because salmon entails fish. However hyponyms in language have sometimes imprecise relations. For example dog is a type of pet, but not all dogs are pets. We would still hope to capture these almost hyponyms though since in most cases when people talk about dogs they talk about them as pets. [4] introduces an approximate notion of entailment called k-hyponymy, which we outline below.

3.3.1 k-Hyponymy

We begin by defining a partial order on positive operators due to Löwner [20].

Definition 3.3.1. (Löwner Order) Let A and B be positive operators.

$$A \sqsubseteq B \text{ if } B - A \text{ is positive.}$$

Proposition 3.3.1. The Löwner order, $\sqsubseteq$, is a partial order on positive operators.

Proof. Let A, B, C be positive operators.

- Reflexivity: $A - A = 0$, which is positive, so $A \sqsubseteq A$.
- Anti-Symmetry: $A \sqsubseteq B \iff 0 \sqsubseteq B - A \iff A - B \sqsubseteq 0$. If we also have $B \sqsubseteq A$ then $A - B \sqsubseteq 0 \sqsubseteq A - B \iff 0 = A - B \iff A = B$.
- Transitivity: If $A \sqsubseteq B \sqsubseteq C$ then $C - A = C - B + B - A$ is a sum of two positive operators. Therefore $0 \sqsubseteq C - A \iff A \sqsubseteq C$

□

Definition 3.3.2. (k-Hyponymy) For positive operators A and B, and $k \in (0, 1]$,

$$A \preceq_k B \text{ if } 0 \sqsubseteq B - kA$$

In this case we say A is a k-hyponym of B or conversely B is a k-hypernym of A.

In the case that A is not a hyponym of B, we will sometimes use the notation $A \preceq_0 B$. Note that the case $k = 1$ corresponds exactly to the Löwner order.

Proposition 3.3.2. If $A \preceq_k B$, then there is necessarily a maximal such value $k_{max} \in (0,1]$ such that $A \preceq_{k_{max}} B$.

[4] gives an expression for calculating the k-max value given two positive self-adjoint matrices.

Proposition 3.3.3. The following properties of k-hyponymy hold:

1. $A \preceq_1 A$ for all positive operators A.

2. k-hyponymy is neither symmetric nor antisymmetric.

3. If $A \preceq_p B$ and $B \preceq_q C$, then $A \preceq_{pq} C$. Note that pq is a lower bound on the hymponymy strength between A and C, and not the maximal such value.

Proof.

1. $A \preceq_1 A$ is exactly reflexivity of $\sqsubseteq$.

2. Consider positive operators in $\mathbb{R}^2$ with basis $\{|0\rangle, |1\rangle\}$.

$$|1\rangle \preceq_1 |0\rangle + |1\rangle \text{ but } |0\rangle + |1\rangle \npreceq_k |1\rangle$$

so k-hyponymy is not symmetric.

$$\frac{1}{3}|0\rangle + \frac{2}{3}|1\rangle \preceq_{1/2} \frac{2}{3}|0\rangle + \frac{1}{3}|1\rangle \text{ and } \frac{2}{3}|0\rangle + \frac{1}{3}|1\rangle \preceq_{1/2} \frac{1}{3}|0\rangle + \frac{2}{3}|1\rangle$$

so k-hyponymy is not anti-symmetric.

3. By assumption $pA \sqsubseteq B$, so $0 \sqsubseteq C - qB \sqsubseteq C - qpA \implies A \preceq_{pq} C$.

□

As an example in our linguistic models say we represent *pet* by the density matrix:

$$[\text{pet}] = \frac{2}{5}|\text{dog}\rangle\langle\text{dog}| + \frac{2}{5}|\text{cat}\rangle\langle\text{cat}| + \frac{1}{5}|\text{gold fish}\rangle\langle\text{gold fish}|$$

Then we have that dog and cat are 2/5-hyponyms of pet and gold fish is a 1/5-hyponym of pet since dogs and cats are more prototypical examples of pets than goldfish.

3.3.2 k-hyponymy for Positive Sentences

To see how entailment works in the logical sense, and in particular how it will interact with negation as defined in the next section it is important to see how k-hyponymy lifts to the level of sentences.

Theorem 3.3.1. Let A, B, C, and D represent nouns such that $A \preceq_p C$ and $B \preceq_q D$, and let $\phi(X \text{ verb } Y)$ denote the meaning of a positive transitive sentence with subject X and object Y. Then,

$$\phi(A \text{ verb } B) \preceq_{pq} \phi(C \text{ verb } D)$$

Proof. From the definition of k-hyponymy, we have: $0 \sqsubseteq C - pA$ and $0 \sqsubseteq D - qB$. Therefore, there exist positive operators ρ_1 and ρ_2 such that:

$$C = pA + \rho_1 \qquad D = qB + \rho_2$$

Applying the sentence reduction maps on $A \otimes [\text{verb}] \otimes B$ and $C \otimes [\text{verb}] \otimes D$, we have:

$$\phi(A \text{ verb } B) = (\epsilon \otimes id_S \otimes \epsilon) \circ (A \otimes [\text{verb}] \otimes B)$$

$$\begin{aligned}
\phi(C \text{ verb } D) &= (\epsilon \otimes id_S \otimes \epsilon)(C \otimes [\text{verb}] \otimes D) \\
&= (\epsilon \otimes id_S \otimes \epsilon)\Big((pA + \rho_1) \otimes [\text{verb}] \otimes (qB + \rho_2)\Big) \\
&= (\epsilon \otimes id_S \otimes \epsilon)\Big((pA \otimes [\text{verb}] \otimes qB) \\
&+ \Big((pA \otimes [\text{verb}] \otimes \rho_2) + (\rho_1 \otimes [\text{verb}] \otimes qB) + (\rho_1 \otimes [\text{verb}] \otimes \rho_2)\Big)\Big) \\
&= pq(\epsilon \otimes id_S \otimes \epsilon)(A \otimes [\text{verb}] \otimes B) \\
&+ (\epsilon \otimes id_S \otimes \epsilon)\Big((pA \otimes [\text{verb}] \otimes \rho_2) + (\rho_1 \otimes [\text{verb}] \otimes qB) + (\rho_1 \otimes [\text{verb}] \otimes \rho_2)\Big) \\
&= pq\phi(A \otimes [\text{verb}] \otimes B) + \rho_3
\end{aligned}$$

Where,

$$\rho_3 = (\epsilon \otimes id_S \otimes \epsilon)\Big((pA \otimes [\text{verb}] \otimes \rho_2) + (\rho_1 \otimes [\text{verb}] \otimes qB) + (\rho_1 \otimes [\text{verb}] \otimes \rho_2)\Big)$$

A, B, ρ_1, ρ_2 are positive operators by assumption, the sentence meaning map is a completely positive map since we are working in CPM(**FHilb**), and p, q are positive scalars. Therefore ρ_3 is a positive operator and we get:

$$\phi(C \text{ verb } D) = pq\phi(A \text{ verb } B) + \rho_3 \implies 0 \sqsubseteq \phi(C \text{ verb } D) - pq\phi(A \text{ verb } B)$$

Therefore, $\phi(A \text{ verb } B) \preceq_{pq} \phi(C \text{ verb } D)$.

□

Note again that in the case of positive transitive sentences above pq is a lower bound and not necessarily the maximum hyponymy strength. The above theorem can also be generalized to any positive sentences with similar structure [4, 3].

3.4 Negation

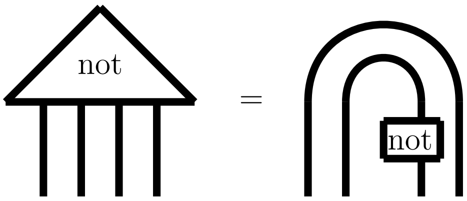
In order to characterize and model logic within linguistic models of meaning, one must have a working model of negation, in particular the word ”not.” It has been an open question for many years how to model negation because, though this seems like an easy task, there are many subtleties in the way we use language that make negation an interesting and difficult thing to define. For example words that are opposites generally have very similar meanings. You might also think that for example, ”not tall” should evaluate to ”short” until you here someone say, ”Yao Ming is not tall, he’s gigantic!”

It has been proposed that orthogonal projection operators, as in quantum logic, could be a good model for negation in the distributional models. Widdows showed in [35] how to use orthogonal projections to allow for search queries such as ”suit NOT lawsuit.” However, words in natural language do not carry meanings in the sense of logics. For example it does not make sense to say that ”red” has a truth value. However, we can assign a truth value to a sentence such as ”Apples are red.” Since we want to be able to use our compositional models, we would hope our model of negation is also compositional. For this reason, our model of negation should also act on the sentence level, not the word level.

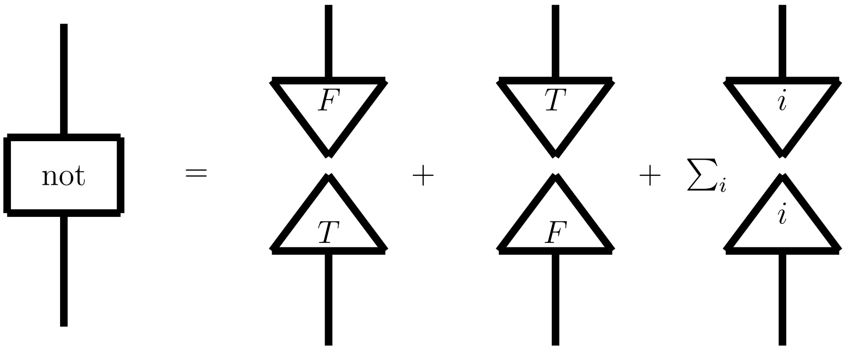
Classification is an important NLP task, and the compositional distributional models have outperformed all previous models in this field. We would hope that our logical models would allow us to retain the classification aspects of our model. This is also important in a full model of meaning because knowing the truth value of a sentence is not necessarily helpful if you cannot say at all what the sentence is referring to. It might be possible as well that a sentence has different truth values based on the context. For example, ”Hillary Clinton is running in 2016,” is a true sentence about the U.S. Presidential Elections, but it is not a true sentence about the Olympic Track and Field competitions in Rio.

To deal with logic in our models we will need a sentence space with true and false components. Let $\{T, F, 1, ..., n\}$ be an orthonormal basis for S where T denotes true, F denotes false, and $i \in [1, n]$ denote the remaining basis vectors of the sentence space, which would correspond to some means of classification, for example sports, politics, etc. If one only cares about the logical aspects of the sentence space, one could just work over the basis $\{T, F\}$.

Definition 3.4.1. We define negation graphically as follows:

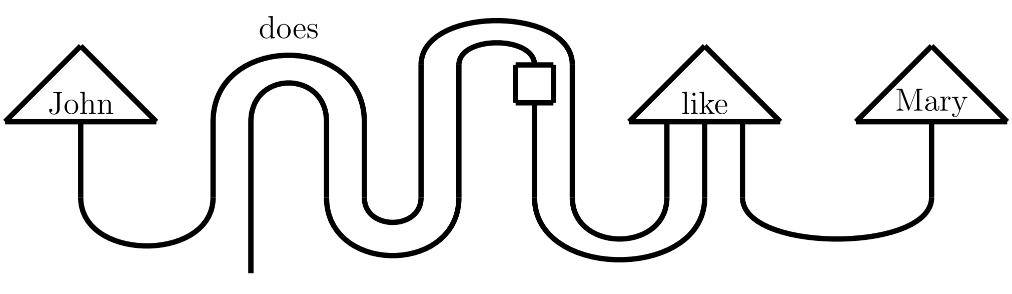


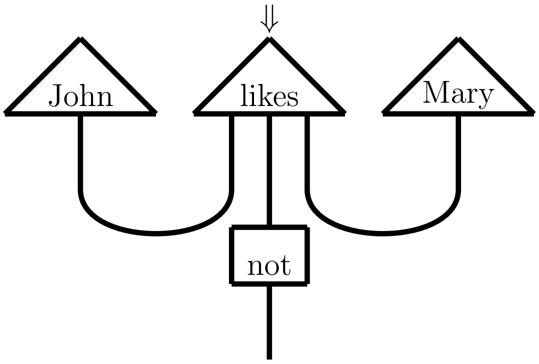
where the not gate is given by:



This negation map acts by swapping the true and false basis elements of the sentence space and keeping all other basis elements the same. This acts on a sentence by projecting the truth value of sentence onto false and the falsity value of the sentence onto true.

The sentence reduction map on a negated sentence acts as in the following example:





To see how negation acts, let's compare a sentence to its negation using k-hyponymy.

Theorem 3.4.1. Let A verb B be a sentence whose meaning $\phi(A \text{ verb } B) = \alpha[T] + \beta[F]$, where α and β are not both 0. Let $\lambda = \min\{\frac{\alpha}{\beta}, \frac{\beta}{\alpha}\}$. Then,

$$\phi(A \text{ verb } B) \preceq_\lambda \phi(A \text{ not verb } B) \qquad \phi(A \text{ not verb } B) \preceq_\lambda \phi(A \text{ verb } B)$$

Proof. The not gate on a sentence acts by swapping the coefficients of $[T]$ and $[F]$ on the sentence meaning space. If $\phi(A \text{ verb } B) = \alpha[T] + \beta[F]$, then $\phi(A \text{ not verb } B) = \beta[T] + \alpha[F]$.

Assume, without loss of generality, that $\alpha \leq \beta$. Then $\lambda = \frac{\alpha}{\beta}$. We have:

$$\phi(A \text{ verb } B) - \lambda\phi(A \text{ not verb } B) = \alpha[T] + \beta[F] - \frac{\alpha}{\beta}(\beta[T] + \alpha[F]) = (0)[T] + (\beta - \frac{\alpha^2}{\beta})[F]$$

Since $\alpha \leq \beta \leq 1$, $\beta - \frac{\alpha}{\beta}\alpha \geq \beta - \alpha \geq 0$, so the above expression is a non-negative sum of positive operators. Therefore

$$0 \sqsubseteq \phi(A \text{ verb } B) - \lambda\phi(A \text{ not verb } B) \iff \phi(A \text{ not verb } B) \preceq_\lambda \phi(A \text{ verb } B)$$

The fact that $\phi(A \text{ verb } B) \preceq_\lambda \phi(A \text{ not verb } B)$ follows similarly.

□

Corollary 3.4.1. If $\alpha = 0$ or $\beta = 0$ then $\phi(A \text{ verb } B) \preceq_0 \phi(A \text{ not verb } B)$.

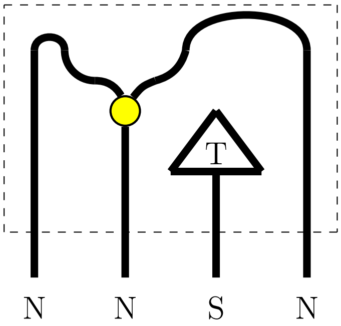
The above corollary represents the case that a sentence is completely true (or false). In this case its negation will be completely false (or true), and the sentences will entail each other to no extent as we would hope. For example if the sentence *John likes Mary* is entirely true then its negation *John does not like Mary* will be false. If we apply negation twice we retrieve the truth value of the original sentence. If ϕ(*John likes Mary*) $= [T]$ then ϕ(*John does not like Mary*) $= [F]$ and ϕ(*John does not not like Mary*) $= [T]$ again.

Corollary 3.4.2. If $\alpha = \beta$ then $\phi(A \text{ verb } B) = \phi(A \text{ not verb } B)$

In this case a sentence is equally as true as it is false, so negation will not change the meaning of the sentence. Say for example that Mary both loves and hates John, then $\phi(\textit{Mary likes John}) = \frac{1}{2}[T] + \frac{1}{2}[F] = \phi(\text{Mary does not like John})$.

3.4.1 Relative Pronouns Revisited

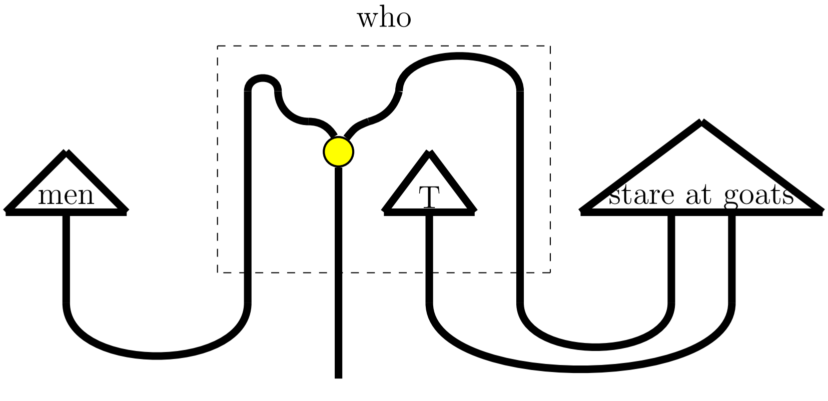
At the end of chapter two, we saw a relative pronoun modelled using Frobenius algebras as in [31, 32], where the sentence space is modelled with just one truth dimension and possibly some dimensions relating to classification. Since our sentence space has both a true and false component to model negation now, we need to modify our relative pronouns in a simple manner to account for this difference. Instead of using the Frobenius unit ζ to delete the sentence space of the verb, we will use the pure truth state $[T] = |T\rangle\langle T|$. For example the relative pronoun, who, becomes:



As an example let $[\text{men}] = \frac{1}{2}|\text{Cane}\rangle\langle\text{Cane}| + \frac{1}{2}|\text{Abel}\rangle\langle\text{Abel}|$. Say Cane stares at goats but Abel does not so that:

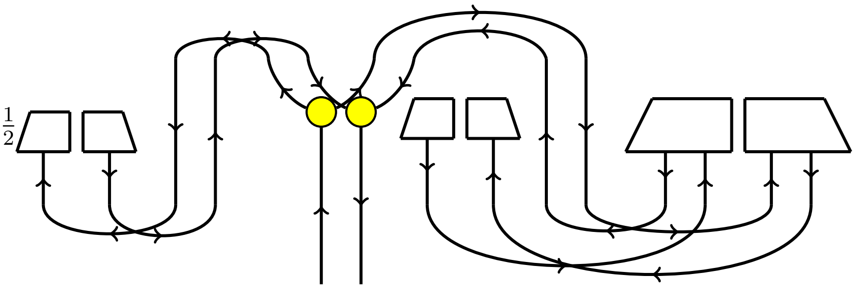
$$\phi(\text{Cane stares at goats}) = [T] \text{ and } \phi(\text{Abel stares at goats}) = [F]$$

.



$$=$$

$$\sum_{\text{Cane, Abel}} \frac{1}{2}$$



This reduces to just $\frac{1}{2}|\text{Cane}\rangle\langle\text{Cane}|$. By comparison, computing the density matrix for ϕ(men who do not stare at goats) gives $\frac{1}{2}|\text{Abel}\rangle\langle\text{Abel}|$. We can then re-normalize the states to remove the lingering 1/2 coefficients.

Chapter 4

Dual Density Matrices: Combining Entailment and Ambiguity to Model Conjunction/Disjunction

Ambiguity and entailment are features of language that often do not appear independently from each other. Some words are general and ambiguous, such as *club*, which could refer to a stick or a group of people, with each meaning being general cases of more specific words, such as golf-club and dance-club. We would like that our models could support both aspects of language. Conjunction and disjunction in natural language also require a level of lexical entailment and ambiguity. Intuitively we can think of *and* as taking an entailment sum over two words or sentences and *or* as creating an ambiguous sum.

In Chapter 3 we saw that Selinger's CPM-construction can be applied to any compact closed category and that the mixed states of CPM(**FHilb**) could be used as a model for either ambiguity or lexical entailment. For any compact category $\mathcal{C}$, CPM($\mathcal{C}$) is a compact category, so we can iterate the CPM-construction to give $\text{CPM}^2(\mathcal{C}) = \text{CPM}(\text{CPM}(\mathcal{C}))$. We will see that states in CPM^2 contain two levels of mixing, and we can use one to model ambiguity and the other for entailment. In [6], Coecke gives an axiomatic description of Selinger's CPM-construction in terms of environment structures and maximally mixed states. [1] generalizes this construction to obtain an axiomatic description of CPM^n.

Previously, CPM(**FHilb**) provided us with the categorical language to discuss and apply density matrices and positive operators. States in CPM^2 are called dual or double density matrices. Dual density matrices and other structures arising in the iterations of CPM have not yet been studied in much detail. Early work on understanding of dual density matrices and their applications in natural language appear in [2, 1], and much of their framework will be presented below.

4.1 Environment Structures

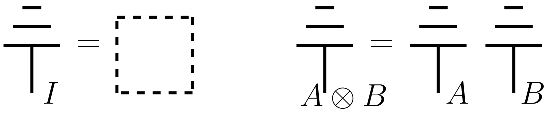
We begin with the mathematical framework of environment structures. In physics and quantum mechanics these structures model maximally mixed states and discarding of quantum systems [8]. In [6] environment structures provide the language to axiomatize Selinger's CPM-construction. Environment structures will be used in this paper to understand CPM^2 and to model conjunction and disjunction.

Definition 4.1.1. (Environment Structure) An environment structure, or $\top$ structure, on a dagger compact closed category $\mathcal{C}$ comprises:

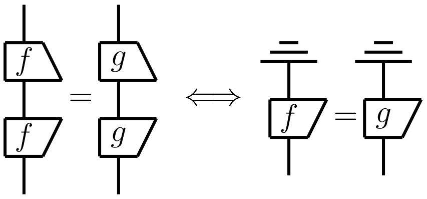
1. A designated state $\top_A$, called the *maximally mixed state*, for each $A \in Ob(\mathcal{C})$ satisfying:

$$\top_I = id_I \qquad \top_{A \otimes B} = \top_A \otimes \top_B$$

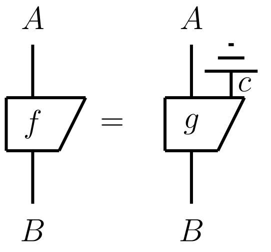
   Or graphically:



2. An all-objects-including sub-dagger compact closed category $\mathcal{C}_\Sigma$ of pure morphisms satisfying:

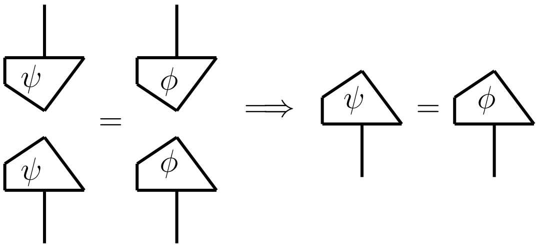


3. A purification operation assigning for each morphism $f \in \mathcal{C}(A, B)$, a pure morphism $g \in \mathcal{C}_\Sigma(A \otimes C, B)$ such that:



Proposition 4.1.1. The *preparation-state agreement axiom* [5] is a special case of axiom 2 in Definition 4.1.1.

Proof. The preparation-state agreement axiom is, graphically:



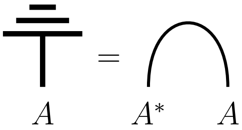
Setting dom(f)=dom(g)=I in axiom 2 gives

$$f \circ f^\dagger = g \circ g^\dagger \implies f \circ \top_I = g \circ \top_I \implies f = g$$

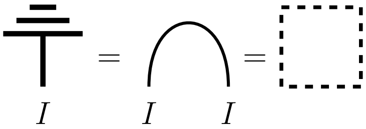
where the last implication follows from axiom 1. □

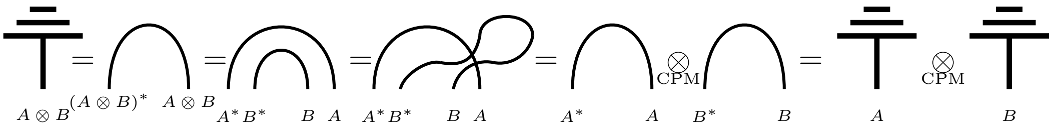
Theorem 4.1.1. If $\mathcal{C}$ is a dagger-compact closed category, then CPM($\mathcal{C}$) has an environment structure.

Proof. Define $\top = \eta_{\mathcal{C}}$:

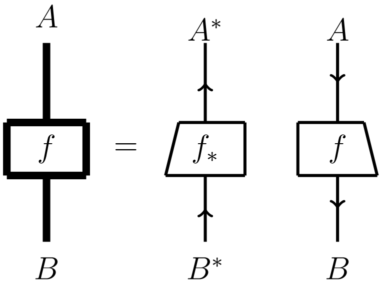


$\top$ satisfies the required properties:

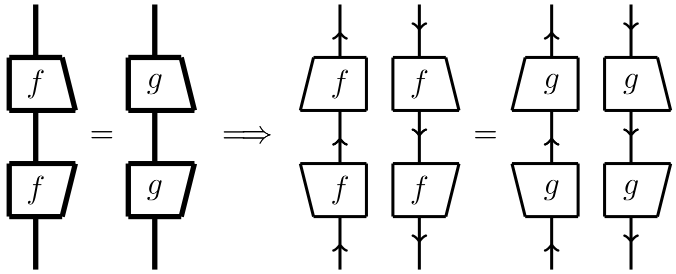




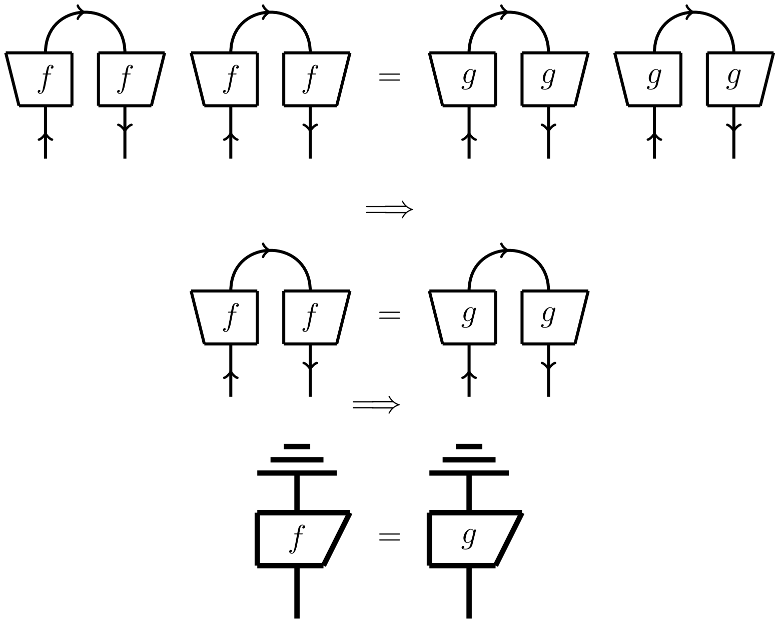
The pure morphisms $f \in \text{CPM}(\mathcal{C})_\Sigma$ are exactly the pure morphisms $f \in \text{CPM}(\mathcal{C})$ given by doubling maps in $\mathcal{C}$:



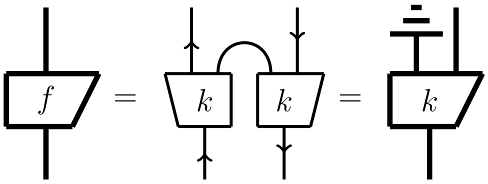
We then have $\forall f, g \in \mathrm{CPM}(\mathcal{C})_\Sigma$:



By bending the inner wires down we get by process state duality into the following equivalent form:



It remains to show that every morphism in $\mathrm{CPM}(\mathcal{C})$ is purifiable, but this is direct from the form of morphisms in $\mathrm{CPM}(\mathcal{C})$ and the definition of $\top$:

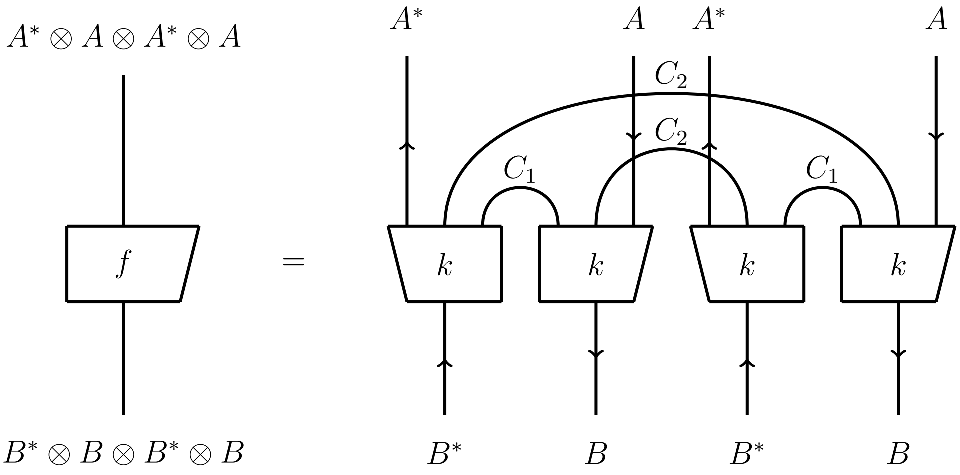


□

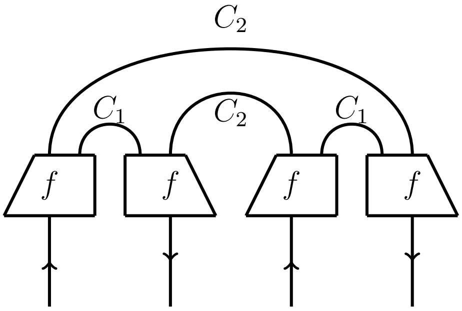
4.2 CPM^2

Recall that for a †-compact closed category, CPM($\mathcal{C}$) had the same objects as $\mathcal{C}$ and its morphism were the completely positive maps of $\mathcal{C}$. CPM^2($\mathcal{C}$) will have the same objects as $\mathcal{C}$ and its morphisms will be completely squared-positive maps.

Definition 4.2.1. Let $\mathcal{C}$ be a †-compact closed category. A morphism $f : A^* \otimes A \otimes A^* \otimes A \to B^* \otimes B \otimes B^* \otimes B$ is completely squared-positive if there exists $C_1, C_2 \in Ob(\mathcal{C})$ and a morphism $k : C_1 \otimes C_2 \otimes A \to B$ such that:

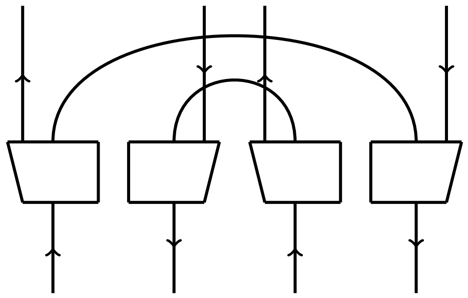

$A^* \otimes A \otimes A^* \otimes A$
f
$B^* \otimes B \otimes B^* \otimes B$
=
A^*
A
A^*
A
C_2
C_2
C_1
C_1
k
k
k
k
B^*
B
B^*
B

In the case $A = I$ we get states of CPM^2($\mathcal{C}$) of the form:

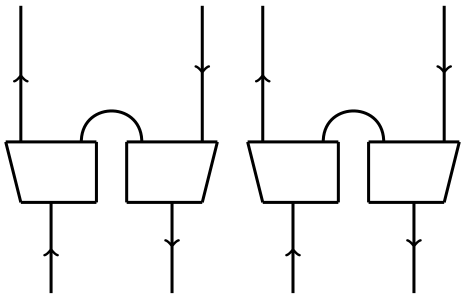


The two ancillary systems, C_1 and C_2, correlate to two different levels of mixing. In our linguistic models this will enable us to represent ambiguity mixing and entailment mixing together in one category. Because the morphisms in CPM^2 have two ancillary systems, we arrive at two notions of purity.

Definition 4.2.2. A completely squared-positive morphism is 1-pure if $C_1 = I$ and 2-pure if $C_2 = I$.



1-pure



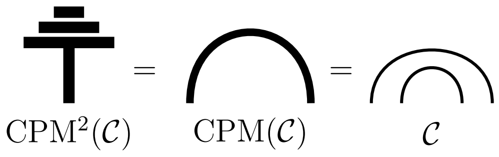
2-pure

We will see linguistically that these notions of purity correspond to an ambiguous word where the homonyms have specific meanings and an unambiguous, general word respectively. A 1,2-pure state will represent an unambiguous, specific word.

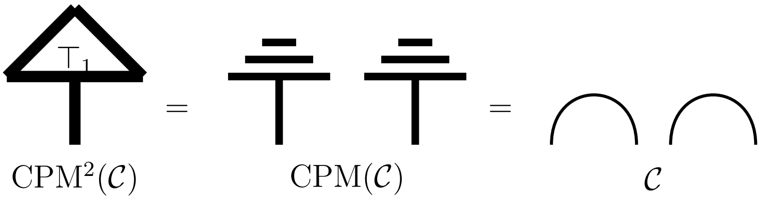
We could have identically characterized the maps of CPM^2 using environment structures. Because $\mathrm{CPM}(\mathcal{C})$ always has an environment structure, $\mathrm{CPM}^2(\mathcal{C})$ has in fact two distinct environment structures, the environment structure related to $\mathrm{CPM}(\mathcal{C})$ and the environment structure of $\mathrm{CPM}(\mathrm{CPM}(\mathcal{C}))$. Just as the caps in $\mathcal{C}$ acted as environment structures in $\mathrm{CPM}(\mathcal{C})$, the caps of $\mathrm{CPM}(\mathcal{C})$, which are the doubled versions of the caps in $\mathcal{C}$, act as environment structures in $\mathrm{CPM}^2(\mathcal{C})$.

Proposition 4.2.1. If $\mathcal{C}$ is a †-compact closed category, then $\mathrm{CPM}^2(\mathcal{C})$ has two environment structures.

Proof. Theorem 4.1.1 applied to $\mathrm{CPM}^2(\mathcal{C})$ means the caps in $\mathrm{CPM}(\mathcal{C})$ are environment structures on $\mathrm{CPM}^2(\mathcal{C})$. That is $\top_2 = \eta_{CPM(\mathcal{C})}$ is an environment structure. Graphically:



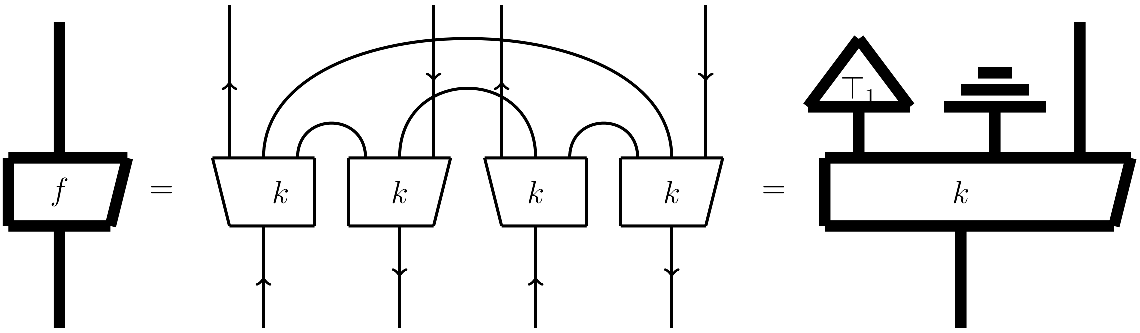
Let $\top = \eta_{\mathcal{C}}$ as in theorem 4.1.1. Then $\top_1 = \top \otimes \top$ is an environment structure.



It follows similarly to the proof of theorem 4.1.1 that $\top_1$ satisfies axioms of an environment structure with the all objects including sub-†-compact closed category $\mathrm{CPM}^2(\mathcal{C})_{\Sigma_1}$ referring to the 1-pure morphisms.

□

In fact, CPM$^2(\mathcal{C})$ satisfies a stronger condition of having a squared-environment structure [1]. In particular, every morphism of CPM$^2(\mathcal{C})$ is 1,2-purifiable:



4.2.1 Dual Density Matrices

Recall that density matrices are positive operators of the form:

$$[\rho] = \sum_i p_i |i\rangle\langle i|$$

where $\{|i\rangle\}_i$ is a set of normalized vectors in a Hilbert space V and $\sum_i p_i = 1$. By process state duality we can turn a density matrix into a vector in $V^* \otimes V$:

$$|\rho\rangle = \sum_i p_i \overline{|i\rangle}|i\rangle$$

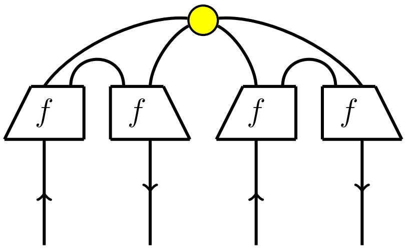
We can then take a probabilistic sum over vector of this form to create a density matrix, or by process state duality a vector in $V^* \otimes V \otimes V^* \otimes V$:

$$[[\Psi]] = \sum_k q_k \overline{|\rho\rangle}|\rho\rangle = \sum_k q_k \overline{((\sum_j p_{jk}\overline{|j_k\rangle}|j_k\rangle))} \otimes (\sum_i p_{ik}\overline{|i_k\rangle}|i_k\rangle))$$

$$= \sum_{ijk} q_k p_{jk} p_{ik} \overline{|j_k\rangle}|j_k\rangle\overline{|i_k\rangle}|i_k\rangle$$

What we have obtained is a dual density matrix. Note in this direct construction of a dual density matrix that there are only three indexes in the sum. An arbitrary state in CPM^2(**FHilb**) has four indexes, two for each ancillary system.

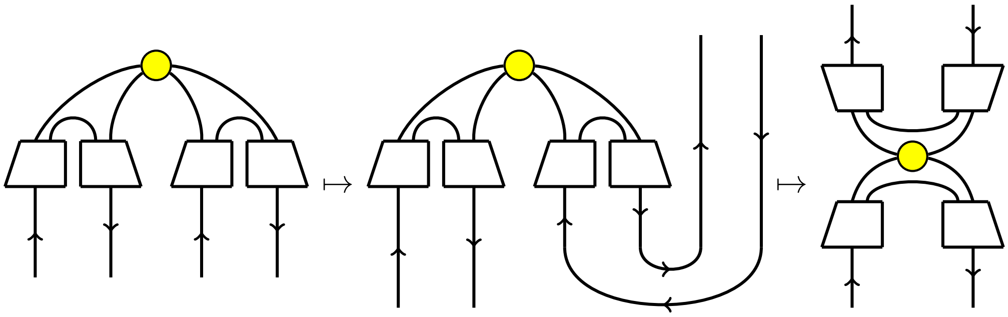
Definition 4.2.3. A dual density matrix is a state of CPM^2(**FHilb**) of the form:

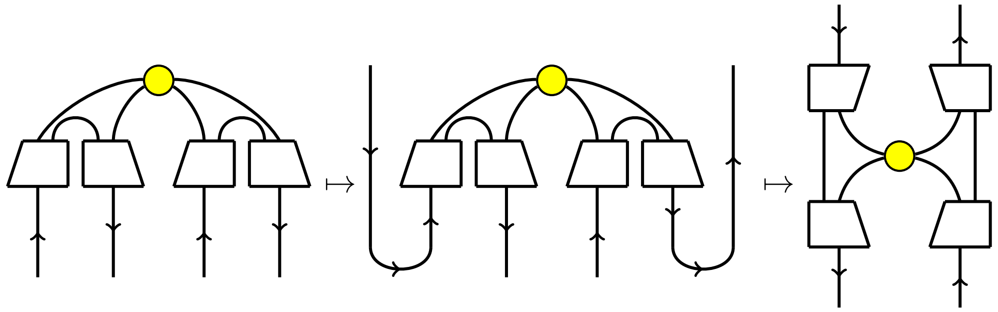


Proposition 4.2.2. The set of dual density matrices is a strict subset of the set of states in CPM^2(**FHilb**)

Proof. The proof of this proposition follows nicely from the graphical calculus and spider fusion rule. The detailed proof, which comes from Maaike Zwart's ongoing doctoral research on dual density matrices, appears in the appendix. □

By process state duality we can turn a state in CPM$^2(\mathcal{C})$ into a completely positive map in CPM$(\mathcal{C})$. In other words we turn a dual density matrix into a positive operator or density matrix. There are two different ways to do this resulting in two different density matrices, hence the name dual density matrices:





We will refer to the top form as density matrix-1 and the bottom as density matrix-2.

4.3 Modeling Ambiguity and Entailment

Consider the word club, an ambiguous word meaning either a place/society or a stick with a heavy end. These two unambiguous meanings entail more specific forms of each kind of club. For example a club-place includes different types of clubs such as sports clubs and dance clubs. A club-stick encompasses the meanings of golf-club as well as a club used as a weapon like a cudgel. Say we take dance-club, sports-club, golf-club, and cudgel as four basis words. We can represent the unambiguous meanings of club (place vs. stick) by the two entailment sums:

$$[club - place] = \frac{1}{2}|\text{dance-club}\rangle\langle\text{dance-club}| + \frac{1}{2}|\text{sports-club}\rangle\langle\text{sports-club}|$$

$$[club - stick] = \frac{1}{2}|\text{golf-club}\rangle\langle\text{golf-club}| + \frac{1}{2}|\text{cudgel}\rangle\langle\text{cudgel}|$$

We can double these terms and sum again to get a full meaning representation for the word club:

$$\begin{aligned}[[\text{club}]] &= \frac{1}{2}(\frac{1}{2}D_*D + \frac{1}{2}S_*S)(\frac{1}{2}D_*D + \frac{1}{2}S_*S) + \frac{1}{2}(\frac{1}{2}G_*G + \frac{1}{2}C_*C)(\frac{1}{2}G_*G + \frac{1}{2}C_*C)\\ &= \frac{1}{8}(D_*DD_*D + D_*DS_*S + S_*SD_*D + S_*SS_*S\\ &+ G_*GG_*G + G_*GC_*C + C_*CG_*G + C_*CC_*C)\end{aligned}$$

In the above notation each letter represents a vector and the asterisk represents its conjugate. For example $D_*DS_*S = \overline{|\text{dance-club}\rangle} \otimes |\text{dance-club}\rangle \otimes \overline{|\text{sports-club}\rangle} \otimes |\text{sports-club}\rangle$.

The reason it is important to do entailment mixing before ambiguity mixing will become apparent in the following section when we consider what the entailment relation should be between conjunction and disjunction.

4.3.1 k-Hyponymy in CPM^2

A completely squared-positive morphism can be turned into a positive operator in two ways. If we use density matrix-1, obtained diagrammatically by bending the right two wires up, to represent dual density matrices we can compare words using k-hyponymy as it has already been defined in the previous chapter. From the above example we would have: [[dance-club]] $\preceq_{1/8}$ [[club]].

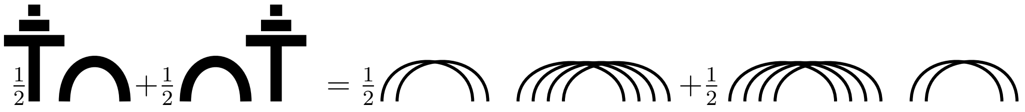
It might cause concern that the coefficients for k-hyponymy are getting quite small. For example we set up the example so that dance-club was a 1/2-hyponym of club-place. Yet, if we compared the doubled versions in CPM^2 we get that: [[dance-club]] $\preceq_{1/4}$ [[club-place]]. However, this is just a side-effect of our examples being quite small. We chose dance-club and sports-club to be distinct basis words, though in a large experiment we would expect the distributional meanings of the words in an entailment sum to have a lot of overlap since they share the features that make them the type of the more general word.

4.4 Conjunction and Disjunction

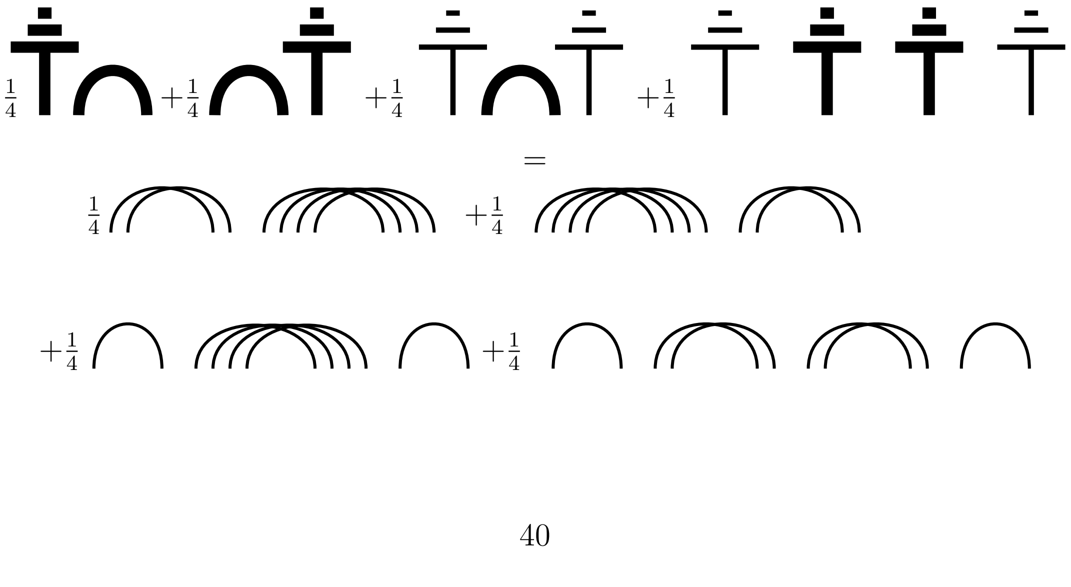
Conjunction and disjunction are two of the main building blocks of any logical theory. In natural language we use the words *and* and *or* as logical connectives. However, as is often the case with language, meaning can be imprecise. For example, *or* sometimes is used as logical disjunction and other times as an exclusive or. If someone asks, "Would you like some tea or coffee?" You could reply, "no" without any confusion, but if you say, "yes", then you would probably receive a confused look and a repeated question, "So... would you like tea *or* coffee?"

Intuitively we can think of *and* as taking an entailment sum over two inputs. For example, if we say, "Dogs and cats are common pets," then we should infer both "dogs are common pets," and "cats are common pets." *Or* on the other hand creates some sense of ambiguity in a phrase/sentence. "Adam or Eve ate the apple," means either "Adam ate the apple," "Eve ate the apple," or "Adam and Eve ate the apple," but in any case we are not sure which. Luckily, we have show already how to model entailment and ambiguity together in the category CPM^2(**FHilb**). Using this intuition we will define *and* and *or* in our compositional distributional models so that they map two inputs onto either an entailment sum or an ambiguity sum. These two distinct sums correspond to the two levels of mixing in CPM^2(**FHilb**) as in the previous sections.

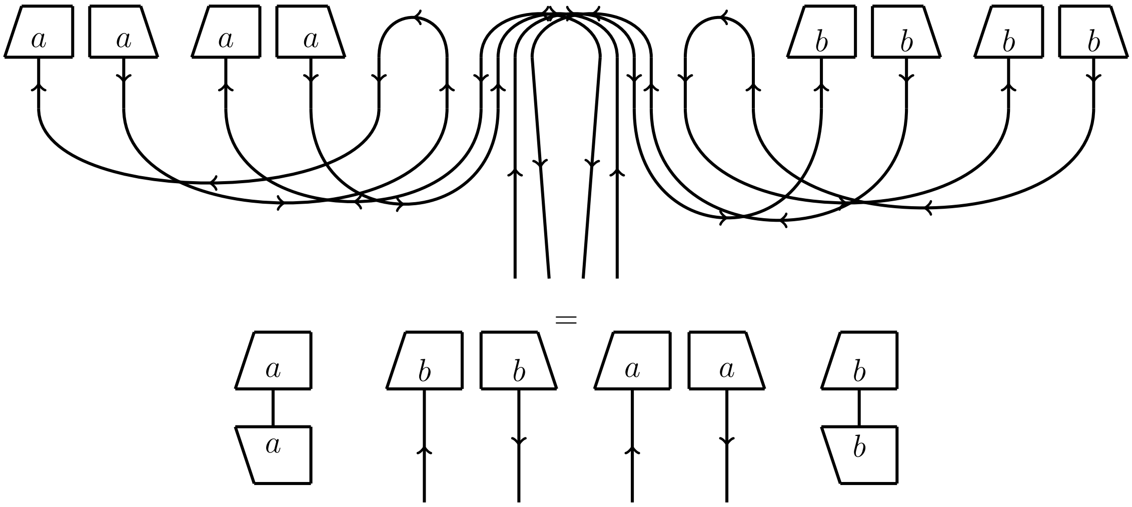
Definition 4.4.1. Define *Or* diagrammatically by the following sum of states:



Define *And* diagrammatically by the following sum of states:



Let's consider how the pregroup reduction acts on a phrase with one of these connectives. Let a, b be nouns represented by pure states. Consider the reduction in the third term of "a and b":



Since words have representations as normalized vectors the two scalars $\langle a|a\rangle$ and $\langle b|b\rangle$ in the above diagram will reduce to 1. The other terms in the connective phrases will reduce similarly giving:

$$\phi(\text{a and b}) = \frac{1}{4}(a_* a a_* a + a_* a b_* b + b_* b a_* a + b_* b b_* b)$$

$$\phi(\text{a or b}) = \frac{1}{2}(a_* a a_* a + b_* b b_* b)$$

which are both dual density matrices. Using the example from the previous section we can see that

[[club]] = $\phi(\phi$(dance-club and sports club) or ϕ(golf-club and cudgel)).

4.4.1 Interaction with Entailment Structure

We can ask what the entailment relation between two noun phrases "a and b" and "a or b" formed from the conjunction and disjunction of two nouns should be. We would hope that "a or b" is a hyponym of "a and b," because if a sentence with "a and b" is true then the sentence with "a or b" will also be true.

Theorem 4.4.1. If a,b are words/phrases of the same type represented by 1,2-pure dual density matrices $[[a]] = a_* a a_* a$, $[[b]] = b_* b b_* b$ in CPM^2(**FHilb**). Then, we have: $\phi(\text{a or b}) \preceq_{\frac{1}{2}} \phi(\text{a and b})$.

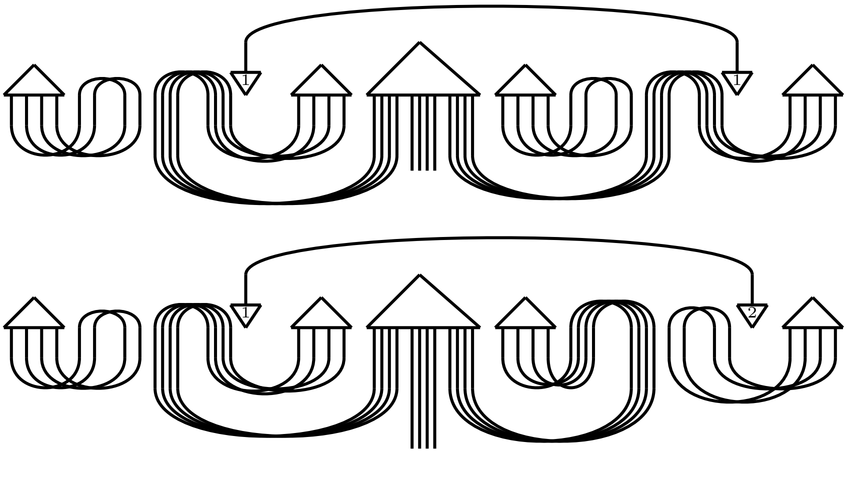
Proof. Applying the sentence reduction maps to the phrase "a and b" gives:

$$\phi(\text{a and b}) = \frac{1}{4}(a_*aa_*a + a_*ab_*b + b_*ba_*a + b_*bb_*b) = \frac{1}{2}\phi(\text{a or b}) + \frac{1}{4}(a_*ab_*b + b_*ba_*a)$$

$a_*ab_*b + b_*ba_*a$ is positive, so $0 \sqsubseteq \phi(\text{a and b}) - \frac{1}{2}\phi(\text{a or b})$. □

4.4.2 "Respectively"

A common occurrence in natural language is a sentence of the form, "a and b are c and d, respectively." We can treat conjunctions of this form in a special way by adjoining an indexing state to the different elements of the sum in the word "and." Connecting the relevant indexing states in the sentence reduction will retain the parts of the sentence corresponding to "a is b" and "c is d" but will send "a is c" and "b is d" to zero. Below is an example of two of the terms in the reduction for ϕ(a and b are c and d respectively). The first corresponds to "b is d" and will remain in the final sum because the indices match. The second corresponds to "b is c" and will go to zero because the indices do not match.



4.5 Example

Consider the noun space with basis given by four names John, Bob, Mary, and Alice. Let the three verbs *likes*, *dislikes*, and *is ambivalent towards* be defined by the following matrices, where the rows represent who the subject likes and the columns represent who likes the object.

$$\text{Likes} = \begin{array}{c} John \\ Bob \\ Mary \\ Alice \end{array} \begin{pmatrix} T & F & T & F \\ T & T & T & T \\ T & T & T & F \\ F & F & F & F \end{pmatrix}$$

$$\text{Is ambivalent towards} = \begin{array}{c} John \\ Bob \\ Mary \\ Alice \end{array} \begin{pmatrix} F & T & F & F \\ F & F & F & F \\ F & F & F & F \\ F & F & T & F \end{pmatrix}$$

$$\text{dislikes} = \begin{array}{c} John \\ Bob \\ Mary \\ Alice \end{array} \begin{pmatrix} F & F & F & T \\ F & F & F & F \\ F & F & F & T \\ T & T & F & T \end{pmatrix}$$

We define general terms *men* and *women* by the entailment sums:

$$[[\text{men}]] = \frac{1}{4}(J_*JJ_*J + J_*JB_*B + B_*BJ_*J + B_*BB_*B) = \phi([[\text{John}]] \otimes \text{and} \otimes [[\text{Bob}]])$$

$$[[\text{women}]] = \frac{1}{4}(M_*MM_*M + M_*MA_*A + A_*AM_*M + A_*AA_*A)$$

Let's take a look at some example sentences. We start with the simple cases of a sentence that is completely true or false:

$$\begin{aligned} \phi(\text{John likes Mary}) &= TTTT \\ \phi(\text{John dislikes Mary}) &= FFFF \\ \phi(\text{John does not like Mary}) &= FFFF \\ \phi(\text{John likes Mary}) &\preceq_0 \phi(\text{John dislikes Mary}) \\ \phi(\text{John dislikes Mary}) &\preceq_1 \phi(\text{John does not like Mary}) \end{aligned}$$

We can see from these sentences the relation between a negation and a negative verb in a simple case. Here we have that *John likes Mary* does not entail at all *John dislikes Mary* or *John does not like Mary.* However, *John dislikes Mary* completely entails *John does not like Mary.* We can also compare connected sentences using our conjunctions and disjunctions. Again in the simple case of sentences whose truth value is completely true or false:

$$\begin{aligned}
\phi(\text{John is ambivalent towards Bob}) &= TTTT \\
\phi(\text{John does not like Bob, and John does not dislike Bob}) &= TTTT \\
\phi(\text{John likes Bob, or John is ambivalent towards Bob}) &= \frac{1}{2}(TTTT + FFFF) \\
\phi(\text{John likes Bob, and John is ambivalent towards Bob}) &= \\
\frac{1}{4}(TTTT + TTFF + FFTT + FFFF)& \\
\phi(\text{John is ambivalent towards Bob}) &\preceq_{1/2} \\
\phi(\text{John likes Bob, or John is ambivalent towards Bob})& \\
\phi(\text{John is ambivalent towards Bob}) &\preceq_{1/4} \\
\phi(\text{John likes Bob, and John is ambivalent towards Bob})&
\end{aligned}$$

Notice the difference in the mixing of truth values in the conjunction of sentences versus the disjunction. In the disjunctive sentence we end up with a higher pure truth value because one of the statements is true. The pure truth value is diminished in the conjunctive sentence, because both are statements are not simultaneously true. However, we still retain some sense in the conjunctive sentence that the statement is equally true as it is false, due to one half being completely true and one half being completely false.

Now let's look at some cases where the words in the sentences have some entailment structure:

$$\begin{aligned}
\phi(\text{Men like Mary}) &= TTTT \\
\phi(\text{Men like Alice}) &= \frac{1}{4}(TTTT + TTFF + FFTT + FFFF) \\
\phi(\text{Men like Women}) &= \frac{5}{8}TTTT + \frac{1}{8}(TTFF + FFTT + FFFF) \\
\phi(\text{Men like Mary}) &\preceq_{5/8} \phi(\text{Men like women}) \\
\phi(\text{Men like Alice}) &\preceq_{1/2} \phi(\text{Men like women})
\end{aligned}$$

We can also consider some subsets of the sentence *men like women* by using "respectively":

$$
\begin{aligned}
&\phi(\text{John and Bob like Mary and Alice, respectively}) = TTTT\\
&\phi(\text{John and Bob like Mary and Alice, respectively}) \preceq_{5/8} \phi(\text{Men like women})
\end{aligned}
$$

We can also compare the conjunction of two nouns in the sentence to the disjunction of two nouns in a sentence:

$$
\begin{aligned}
\phi(\text{Mary and Alice like men}) &= \frac{1}{4}(TTTT + TTFF + FFTT + FFFF)\\
\phi(\text{Mary or Alice like men}) &= \frac{1}{2}(TTTT + FFFF)\\
\phi(\text{Mary or Alice like men}) &\preceq_{1/2} \phi(\text{Mary and Alice like men})
\end{aligned}
$$

A better example of how conjunctions and disjunctions operate between nouns is to consider them inside noun phrases using relative pronouns:

$$
\begin{aligned}
\phi(\text{Men who like Mary}) &= [[\text{men}]]\\
\phi(\text{Men who like Alice}) &= [[\text{Bob}]]\\
\phi(\text{Men who like Mary or Alice}) &= \frac{1}{5}(J_*JJ_*J + J_*JB_*B + B_*BJ_*J + 2B_*BB_*B)\\
\phi(\text{Men who like women}) &= \frac{1}{9}(J_*JJ_*J + 2J_*JB_*B + 2B_*BJ_*J + 4B_*BB_*B)\\
[[\text{Bob}]] &\preceq_{1/4} [[\text{men}]]\\
[[\text{Bob}]] &\preceq_{2/5} \phi(\text{Men who like Mary or Alice})\\
[[\text{Bob}]] &\preceq_{4/9} \phi(\text{Men who like Mary and Alice})
\end{aligned}
$$

We already had that *Bob* was a hyponym of *men*, which was an equal mixture of *John and Bob*. However, Bob becomes a stronger hyponym of the phrase *Men who like Mary or Alice* because John only likes Mary, where as Bob likes Mary and Alice. The hyponymy strength increases again between *Bob* and *men who like Mary and Alice* because *Bob likes Mary and Alice* is completely true where as *John likes Mary and Alice* is not.

Chapter 5

Conclusions

In this dissertation I have addressed the main elements for logic within compositional distributional models of meaning. Building on the models of [10], we applied Selinger's CPM construction to model lexical entailment and ambiguity as in [4, 26]. In CPM(**FHilb**) we could model words with ambiguous or general meanings by passing from the vector representations of **FHilb** to density matrices and positive operators. We defined a negation operation to model *not* at the sentence level, and compared sentences to their negations using k-hypnoymy as an entailment relation. Iterating the CPM-construction brought us to the category CPM^2(**FHilb**) and we transitioned from representing word meanings as density matrices to using dual density matrices. Dual density matrices have two levels of mixing, so we could model words with both ambiguous and general meanings in one dual density matrix. We saw that CPM($\mathcal{C}$) has an environment structure and that CPM^2($\mathcal{C}$) has two environment structures. Using these two different environment structures we were able to build maps that took two different kinds of sums over two dual density matrices of the same type. These two kinds of sums correlated to the two levels of mixing, one for ambiguity and one for entailment. We used these two distinct sums to model *and* and *or* within the compositional distributional models. Additionally, we found a relation between these models for conjunction and disjunction with *or* being a hyponym of *and*. Finally, we presented some examples that combined conjunction, disjunction, and negation and compared the entailment relation on sentences and phrases of various constructions. We saw that these constructions behaved in a way we would expect for logical sentences in natural language with an approximate entailment relation.

5.1 Future Work

Defining the main elements for logic is only the beginning steps towards fully understanding and formalizing logic in linguistic models. The examples presented in this paper were limited and primarily for proof of concept. Now that we have a model for these logical elements a full scale experiment needs to be done with data taken from corpora. Large scale experiments must prove successful and efficient enough before applying these models to natural language processing tasks. Dual density matrices provide a lot of extra structure but they will also be more computationally expensive, so it will be important to see in an experiment whether the extra linguistic power is worth the extra computational resources. CPM can also be iterated as many times as we like, CPM^n. This could allow for modelling increasingly many features of language. CPM($\mathcal{C}$) was clearly richer than $\mathcal{C}$ and CPM^2 was clearly richer than CPM. It is not clear though, and is an area of ongoing research, whether this increase in expressive power continues indefinitely or not. Another main area of linguistic modelling uses convex regions in conceptual spaces to model words. It will be interesting to see if the elements of logic defined in this paper will carry over to these other types of categories.

Chapter 6

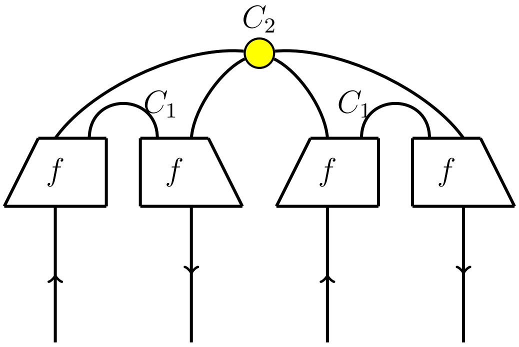
Appendix

6.1 Proof of Proposition 4.2.2

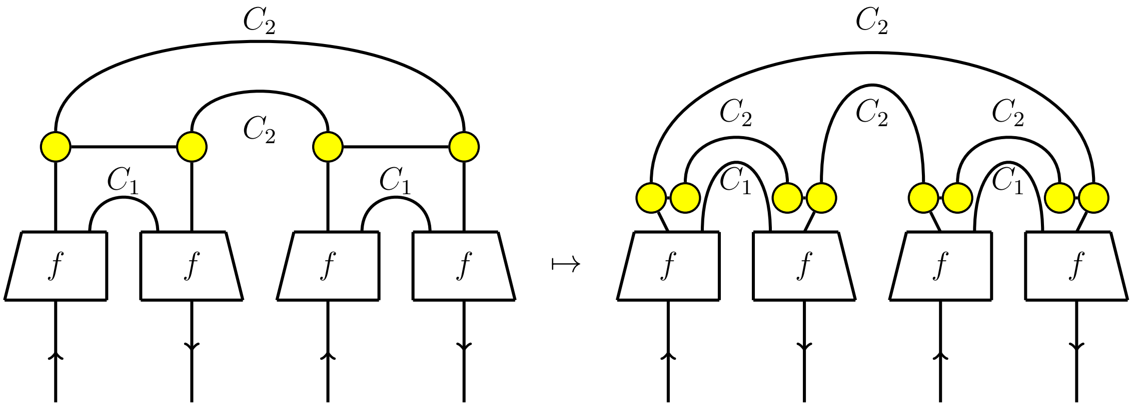
The set of dual density matrices is a strict subset of the set of states in CPM^2(**FHilb**):

First we show that every dual density matrix is a state in CPM^2(**FHilb**).

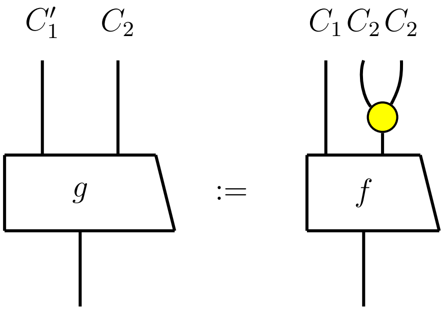
A dual density matrix is of the form:



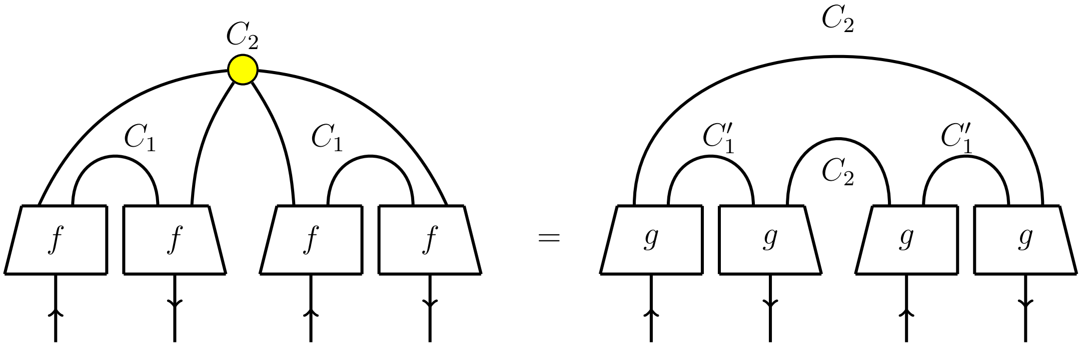
Using the reverse of the spider fusion rule and topological manipulations allowed in the graphical calculus we can redraw our density matrices as:



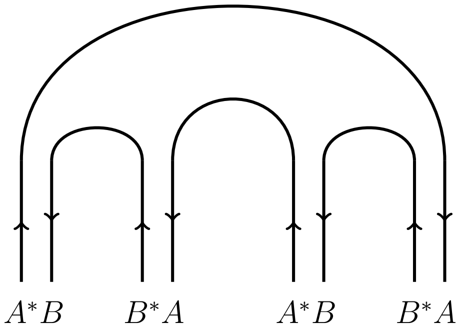
Letting $C_1' = C_1 \otimes C_2$ and $g : C_1' \otimes C_2 \to B$ be:



we have:



The fact that the set of dual density matrices is a strict subset is because there are states in CPM^2(**FHilb**) that are not dual density matrices as we have defined them. The following map is completely squared-positive, but is cannot be written in the form of a dual density matrix:



Bibliography

[1] Daniela Ashoush. Categorical models of meaning: Accommodating for lexical ambiuity and entailment. Master's thesis, University of Oxford, St. Hugh's College, Oxford, 2015.

[2] Daniela Ashoush and Bob Coecke. Dual density operators and natural language meaning. *EPTCS*, 221, 2016.

[3] Desislava Bankova. Comparing meaning in language and cognition: P-hyponymy, concept combination, asymmetric similarity. Master's thesis, University of Oxford, Oxford, 2015.

[4] Desislava Bankova, Bob Coecke, Martha Lewis, and Daniel Marsden. Graded entailment for compositional distributional semantics. *CoRR*, abs/1601.04908, 2016.

[5] Bob Coecke. De-linearizing linearity: Projective quantum axiomatics from strong compact closure. *Electronic Notes in Theoretical Computer Science*, 170:49–72, 2007.

[6] Bob Coecke. Axiomatic description of mixed states from selinger's cpm-construction. *Electronic Notes in Theoretical Computer Science*, 210:3–13, 2008.

[7] Bob Coecke and Matty J. Hoban, editors. *Proceedings of the 10th International Workshop on Quantum Physics and Logic, QPL 2013, Castelldefels (Barcelona), Spain, July 17-19, 2013*, volume 171 of *EPTCS*, 2014.

[8] Bob Coecke and Aleks Kissinger. Picturing quantum processes. 2016.

[9] Bob Coecke and Simon Perdrix. *Environment and Classical Channels in Categorical Quantum Mechanincs*, pages 230–244. Volume 6247 of Dawar and Veith [11], 2010.

[10] Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. Mathematical foundations for a compositional distributional model of meaning. *CoRR*, abs/1003.4394, 2010.

[11] Anuj Dawar and Helmut Veith, editors. *Computer Science Logic: 24th International Workshop, CSL 2010, 19th Annual Conference of the EACSL, Brno, Czech Republic, August 23-27, 2010. Proceedings*, volume 6247. Springer Berlin Heidelberg, 2010.

[12] Ellie D'Hondt and Prakash Panangaden. Quantum weakest preconditions. *Mathematical Structures in Computer Science*, 16(3):429–451, 2006.

[13] Edward Grefenstette and Mehrnoosh Sadrzadeh. Experimental support for a categorical compositional distributional model of meaning. *CoRR*, abs/1106.4058, 2011.

[14] Dimitri Kartsaklis. Coordination in categorical compositional distributional semantics. In Kartsaklis et al. [16], pages 29–38.

[15] Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Stephen Pulman. A unified sentence space for categorical distributional-compositional semantics: Theory and experiments. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING): Posters*, pages 549–558, Mumbai, India, December 2012.

[16] Dimitrios Kartsaklis, Martha Lewis, and Laura Rimell, editors. *Proceedings of the 2016 Workshop on Semantic Spaces at the Intersection of NLP, Physics and Cognitive Science*, volume 221 of *EPTCS*, 2016.

[17] J. Lambek. *Type Grammar Revisited*, pages 1–27. Volume 1582 of Lecomte et al. [19], 1999.

[18] Joachim Lambek. The mathematics of sentence structure. *The American Mathematical Monthly*, 65(3):154–170, 1958.

[19] Alain Lecomte, Francois Lamarche, and Guy Perrier, editors. *Logical Aspects of Computational Linguistics*, volume 1582 of *Lecture Notes in Computer Science.* Springer Berlin Heidelberg, 1999.

[20] K. T. Löwner. Über monotone matrixfunktionen. *Mathematische Zeitschrift*, 38:177–216, 1934.

[21] Saunders MacLane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1978.

[22] Daniel Marsden and Jamie Vicary. Categorical quantum mechanics: An introduction, 2016.

[23] J. Von Neumann. *Mathematische Grundlagen Der Quantenmechanik*. Springer-Verlag, 1932. Translation :[24].

[24] J. Von Neumann. *Mathematical Foundations of Quantum Mechanics*. Princeton University Press, 1955.

[25] Robin Piedeleu. Ambiguity in categorical models of meaning. Master's thesis, University of Oxford, Balliol College, Oxford, 2014.

[26] Robin Piedeleu, Dimitri Kartsaklis, Bob Coecke, and Mehrnoosh Sadrzadeh. Open system categorical quantum semantics in natural language processing. *CoRR*, abs/1502.00831, 2015.

[27] Tamara Polajnar. Collaborative training of tensors for compositional distributional semantics. *CoRR*, abs/1607.02310, 2016.

[28] Anne Preller. From logical to distributional models. In Coecke and Hoban [7], pages 113–131.

[29] Anne Preller and Joachim Lambek. Free compact 2-categories. *Mathematical Structures in Computer Science*, 17(02):309–340, 2007.

[30] Anne Preller and Mehrnoosh Sadrzadeh. Bell states and negative sentences in the distributed models of meaning. *Electronic Notes in Theoretical Computer Science*, 270(2):141–153, 2011.

[31] Mehrnoosh Sadrzadeh, Stephen Clark, and Bob Coecke. The frobenius anatomy of word meanings I: subject and object relative pronouns. *CoRR*, abs/1404.5278, 2014.

[32] Mehrnoosh Sadrzadeh, Stephen Clark, and Bob Coecke. The frobenius anatomy of word meanings II: possessive relative pronouns. *CoRR*, abs/1406.4690, 2014.

[33] Hinrich Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

[34] Peter Selinger. Dagger compact closed categories and completely positive maps. *Electronic Notes in Theoretical Computer Science*, 170:139–163, 2007.

[35] Dominic Widdows and Stanley Peters. Word vectors and quantum logic: Experiments with negation and disjunction. *Mathematics of Language*, 8:141–154, 2003.