Conversational Negation in Compositional Distributional Semantics

Candidate no. 1055080

Word count: 21976

A thesis submitted for the degree of

MSc in Computer Science

Trinity 2021

Acknowledgements

I want to thank my supervisor[1] for introducing me to this fascinating field and their fantastic supervision. They helped me besting challenges and knew what to say when I struggled. I would especially like to thank the two co-authors with whom we published two papers on conversational negation. Without them, this thesis would not be where it is today, and the process of getting there would certainly not have been as fun.

I am incredibly grateful to my parents, who supported my academic career. They did this with great advice and by proofreading applications and theses. However, most importantly, by raising me to ask questions and see the joy in mathematics. I want to thank HK, FR and SF for helping me make this thesis more understandable to an informed audience by helping me identify complicated parts. In addition, I want to thank MR for their diligent proofreading of all my work. Finally, I would like to thank everyone who informed this thesis through our relevant discussions.

[1]To comply with exam regulations requiring me to hide my identity, I have removed all names from the acknowledgement section. The final version of this thesis contains the names of the people I would like to thank.

Abstract

Negation in conversation — conversational negation — is inherently difficult to model. It does not follow straightforward rules such as negation in mathematics. Instead of just denying information, it additionally elicits alternatives. This process builds on the listeners understanding of the negation and its context. This thesis will propose a series of frameworks to model conversational negation in compositional, distributional semantics, particularly in DisCoCirc. We will grow the scope of the negation from (1) individual words to (2) multiple words to (3) evolving meanings to (4) sentences, each step building on the previous.

For the negation of individual words, we propose and experimentally validate multiple operations. Building on psychological observations, these negations model information denial using logical negation. However, they additionally capture the listeners understanding of the world to derive the contexts in which a word usually appears. These contexts are utilised to correct the result of the logical negation to match the human intuition of conversational negation. Following psychological experiments, we grow the scope of conversational negation by composing the negations of individual words to model the negation of multiple words. Thus, the negation of a sentence becomes a negation of a subset of its parts. Here, the context informs the correct interpretation of the negation. Expanding to evolving meanings, which are inherent to DisCoCirc, we propose a third framework. This conversational negation of evolving meanings acts upon the update structure of the negated sentence. Once more, the context informs the correct interpretation of the negation. Finally, we combine the negation frameworks to propose an all-encompassing conversational negation operation for sentences in DisCoCirc.

Contents

The meaning of "no"

1 Introduction

Contents

1.1 Negation in conversation

Negation in natural language, conversational negation, is intrinsic to all human languages. It differentiates human communication from that of other animals (Horn, 1972). However, it is complicated. Conversational negation does not follow clear cut rules such as negation in mathematics. For millennia, philosophers, logicians, psychologists, and linguists have tried to understand the deceptively simple-looking word `no`. There are many differing views on interpreting negation, ranging back as far as Plato (Lee, 1972).

This MSc thesis will propose a framework that models conversational negation of sentences in compositional, distributional semantics. For this, we will start by modelling the negation of words, grow the scope to multiple words and evolving meanings and finally combine our proposals to model conversational negation of sentences.

We will rely on the interpretation of conversational negation presented by Prado and Noveck (2006). They identify two diametrically opposed views; the *narrow view* and the *search-for-alternatives-view*. On the one hand, the *narrow view* proposes

that negation's sole function is to deny a proposition (Evans, 1996). On the other hand, psychologists such as Oaksford and Stenning (1992) view negation not only as a simple denial of information but rather as eliciting alternatives in the listener's mind. In their *search-for-alternatives-view*, humans seek to interpret a negation by considering possible alternatives. Prado and Noveck (2006) propose a combination of both views, in that humans initially view negation as information denial but later embark on a search for alternatives, if necessary. This search for alternatives, when it occurs, marks one key difference between conversational negation in language and logical negation in mathematics. Let us consider the following two sentences:

a) `This is not a chicken; this is a goose.`

b) `This is not a chicken; this is a spaceship.`

Despite both sentences being grammatically and logically correct, the latter seems unusual. The alternatives to the word `chicken`, elicited in the listener's mind, seem to include `goose` but not `spaceship`. Therefore the second sentence defies our expectations.

We observe that these alternatives, elicited by the negation, are inherently context dependent (Kruszewski et al., 2016). If we talk about `not chicken` in the context of `animals`, we might refer to `geese`, while in the context of `meat`, we might instead talk about `beef`. Even the alternative of `spaceship` might be plausible under the right 'contextual pressure' (Oaksford & Stenning, 1992). For example, in some prequel to Star Trek, the spaceships might have rather unusual cloaking mechanisms.

Therefore, the core hypothesis at the heart of this thesis is:

conversational negation is context dependent

This hypothesis will guide our frameworks throughout the growing scope of negations we are modelling.

In a second step, we notice that the *search-for-alternatives-view* can be extended to the negation of sentences. Take, for example, the sentence "`Bob did not drive`

`to Oxford by car.`". Following Oaksford and Stenning (1992), under the *search-for-alternatives-view*, we can interpret this statement in multiple ways, including but not limited to:

a) `Bob did not drive to Oxford by car - Alice did`

b) `Bob did not drive to Oxford by car - He drove to Cambridge`

c) `Bob did not drive to Oxford by car - He drove a van`

The negation's interpretation depends not only on the alternatives the listener elicits but also on what they perceive to be the target of the negation (Oaksford & Stenning, 1992) — here marked by being underlined. This is, once more, informed by the context. For example, in a text about Bob's favourite car trip, the destination of the adventure is most likely the target of the negation — `Bob did not drive to Oxford by car.`

This thesis aims to capture these and some additional intuitions about conversational negation. We will propose a framework compatible with the DisCoCirc model by Coecke (2020), a model for categorical, compositional, distributional semantics. The DisCoCirc model provides a method to represent written text as circuits. These circuits represent the interaction of meanings in texts and model how sentences update preconceived notions in our heads. To make our framework for conversational negation compatible with this framework, we additionally have to model the negation of evolving meanings such as actors about whom we learn details throughout a story (see Section 3.4 for an explanation of evolving meaning). We will conclude by providing a framework for conversational negation of sentences.

1.2 Outline of the thesis

To provide the reader with the necessary background, we will summarize the definitions of and graphical representation for monoidal, compact closed categories (Chapter 2). Then we will recapitulate the categorical, compositional, distributional frameworks, going from DisCoCat (Coecke et al., 2010) to DisCoCirc (Coecke, 2020) (Chapter 3).

From there, we will first introduce a framework that captures the conversational negation of words and experimentally validate it (Chapter 4). Next, utilising this framework, we will capture the conversational negation of multiple words (Chapter 5). Then we will explore the negation of dynamically evolving meanings (Chapter 6). Finally, we put the frameworks together to introduce an operation for conversational negation of sentences compatible with the DisCoCirc framework (Chapter 7).

We observe that throughout the thesis, we grow the scope of the negation; from individual words to multiple words to dynamic meanings to sentences. As all these negations build on the same fundamental observation — negation is context dependent — the chapters are structured in a repeating manner, reflecting this growth. In each chapter, we will start with an intuition into the negation, propose a general framework and finish by exploring how the influence of the context could be quantified. Observing this structure will hopefully increase accessibility. It also allows us to compare the different frameworks more easily and to see how they interact.

1.3 Contributions

As part of the research for this MSc thesis, we published two papers; Rodatz et al. (2021) and Shaikh et al. (2021)[1]. Much of the work in those two papers is integrated into this thesis. Beyond the already published papers, findings will be generalised, elaborated upon, and experiments will be extended. Additionally, final steps will be taken towards making the findings applicable to the DisCoCirc framework. The contributions of this thesis are:

1) For conversational negation of words (Chapter 4)

    1a) Propose a framework based on weighted sums - generalising the proposal from Rodatz et al. (2021) thereby providing more freedom for additional exploration and unifying our proposals for converstional negation

[1] I have been in contact with Lucy Traves about the fact that citing our papers may give away my identity. We have agreed that it is more important to cite the papers.

1b) Propose a method to derive these weights - as presented in Rodatz et al. (2021)

1c) Experimentally validate the proposed framework - expanding upon the experiments in Rodatz et al. (2021) by exploring more aspects and proposing new experiments

2) For conversational negation of multiple words (Chapter 5)

2a) Propose a framework based on weighted sums - as presented in Shaikh et al. (2021)

2b) Propose an intuition for a method to derive these weights - as presented in Shaikh et al. (2021)

3) For conversational negation of evolving meanings (Chapter 6)

3a) Propose a framework based on weighted sums - a new framework that overcomes the challenges of the proposal in Shaikh et al. (2021)

3b) Propose an intuition for a method to derive these weights - utilizing the work presented in Shaikh et al. (2021)

4) For conversational negation of sentences (Chapter 7)

4a) Propose a framework based on weighted sums - thereby tying the proposed frameworks together to make conversational negation applicable to the DisCoCirc framework

A graphical account

2

Categorical introduction

Contents

The branch of mathematics known as *category theory* formalises mathematical structures by solely considering objects and morphisms between those objects.

This chapter will introduce symmetric, compact closed, monoidal categories and their graphical representation (surveyed by Selinger, 2011). Additionally, we will introduce functors between monoidal categories, mapping structure from one category into another. We will utilise this functor by observing that the representations of both grammar and meaning of words are modelled by compact closed categories. This allows us to compose the meaning of words through updates informed by the grammar to derive the meaning of sentences and texts.

We will only give a brief introduction to all the categorical concepts used in this thesis. For a more comprehensive overview, we recommend to refer to Coecke and Paquette (2011) for an introduction to applied category theory and Coecke and Kissinger (2017) for an overview of the graphical language used throughout this thesis.

2.1 Categories

We will begin by defining a category:

Definition 1. *A **category** $\mathcal{C}$ consists of:*

1. *a collection of objects $|\mathcal{C}|$*
2. *for every pair of objects $A, B \in |\mathcal{C}|$, a collection of morphisms $\mathcal{C}(A, B)$ from A to B (we often write $f \in \mathcal{C}(A, B)$ as $f : A \to B$)*
3. *for any three objects $A, B, C \in |\mathcal{C}|$, a composition operation $- \circ - : \mathcal{C}(A, B) \times \mathcal{C}(B, C) \to \mathcal{C}(A, C)$ such that any two morphisms $f \in \mathcal{C}(A, B), g \in \mathcal{C}(B, C)$ can be composed to $g \circ f \in \mathcal{C}(A, C)$ (read "g after f")*
4. *for every object $A \in \mathcal{C}$, an identity morphism $id_A : A \to A$*

such that the composition operation is associative and the identity morphism is the unit, that is:

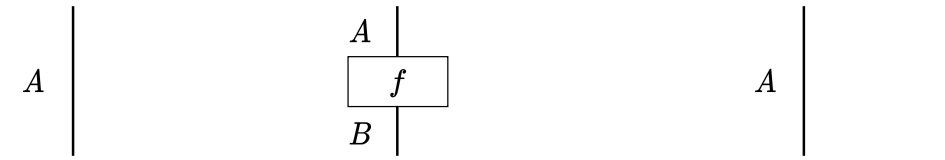
1. *for any three morphisms $f : A \to B, g : B \to C, h : C \to D$, we have:*

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

2. *for all morphisms $f : A \to B$ we have:*

$$f \circ id_A = f = id_B \circ f$$

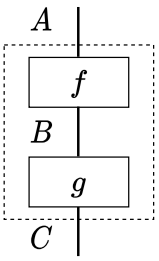
In the graphical language for categories, objects are represented as wires and morphisms are represented as boxes. We thus have (reading from top to bottom):



Object A Morphism $f : A \to B$ Identity $id_A : A \to A$

The identity morphism gets a special representation as a naked wire. It thus looks identical to the representation of an object. Intuitively the identity morphism leaves the object untouched.

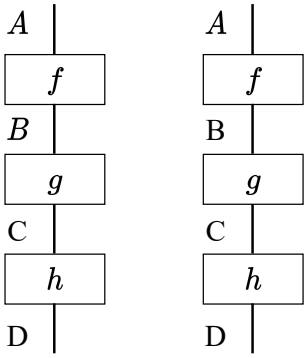
We can compose morphisms by putting them after one another:



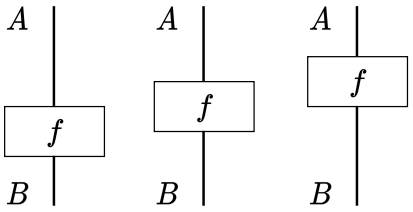
$g \circ f : A \to C$

where the dotted box is the new morphism $g \circ f$ from A to C. The dotted box is usually not drawn in the diagrammatic representation.

Associativity and unitality are thus intrinsic to our representation, we have:



$h \circ (g \circ f) \;=\; (h \circ g) \circ f$



$f \circ id_A \;=\; f \;=\; id_B \circ f$

We can omit the dotted box due to associativity. The unitality of the identity allows us to move morphisms along a wire, as long as they do not pass another morphism.

This simple example illustrates one of the major appeals of the diagrammatic representation; complicated equalities and proofs become intuitive. Instead of relying on an equation to enforce equality between expressions, equal expressions are naturally the same under our representation. This effect is much more pronounced once equations become larger.

2.2 Monoidal categories

While categories in general allow us to compose morphisms sequentially, monoidal categories allow us to also compose morphisms in parallel.

Definition 2 (Heunen and Vicary, 2018, Definition 1.1)**.** *A* ***monoidal category*** $\mathcal{C}$ *is a category equipped with:*

1. *a bifunctor* $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$*, called the tensor product. This bifunctor assigns each pair of objects* $A, B \in |\mathcal{C}|$ *to a composite object* $A \otimes B$ *and each pair of morphisms* $f : A \to B, g : C \to D$ *to a parallel composition* $f \otimes g : A \otimes C \to B \otimes D$
2. *a unit object* $I \in |\mathcal{C}|$
3. *for any three objects* $A, B, C \in |\mathcal{C}|$*, a natural isomorphism* $\alpha_{A,B,C} : (A \otimes B) \otimes C \to A \otimes (B \otimes C)$ *called the associator*
4. *for any object* $A \in |\mathcal{C}|$*, a natural isomorphism* $\lambda_A : I \otimes A \to A$ *called the left unitor*
5. *for any object* $A \in |\mathcal{C}|$*, a natural isomorphism* $\rho_A : A \otimes I \to A$ *called the right unitor*

The associator and unitors must obey the triangle equations:

$$\forall A, B \in |\mathcal{C}|, \quad \rho_A \otimes id_B = (id_A \otimes \lambda_B) \circ \alpha_{A,I,B}$$

and the pentagon equations:

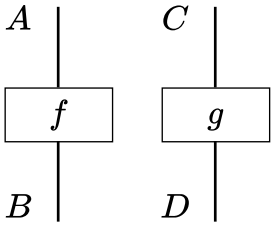
$$\forall A, B, C, D \in |\mathcal{C}|, \quad \alpha_{A,B,C\otimes D} \circ \alpha_{A\otimes B,C,D} = (id_A \otimes \alpha_{B,C,D}) \circ \alpha_{A,B\otimes C,D} \circ (\alpha_{A,B,C} \otimes id_D)$$

The triangle and pentagon equations require α, λ and ρ to interact in an expectable manner[1].

A monoidal category is strict when the associator and unitors are identities. As every monoidal category is monoidally equivalent to a strict monoidal category (Heunen & Vicary, 2018, Theorem 1.38), we will omit the natural isomorphisms and consider them as equalities.

[1]The triangle and pentagon equations have more readable representations as commuting diagrams, as shown in Heunen and Vicary (2018, Equations 1.1, 1.2). We will not elaborate on those here. Instead, we refer the reader to the cited material.

Graphically, the unit is simply not drawn. Parallel composition via the tensor product is drawn as putting the morphism next to each other. We get:



Unit object I $f \otimes g$

Due to the properties of strict monoidal categories, the order of applying the monoidal bifunctor becomes irrelevant. This allows us to draw any number of wires next to each other via multiple applications of the tensor product.

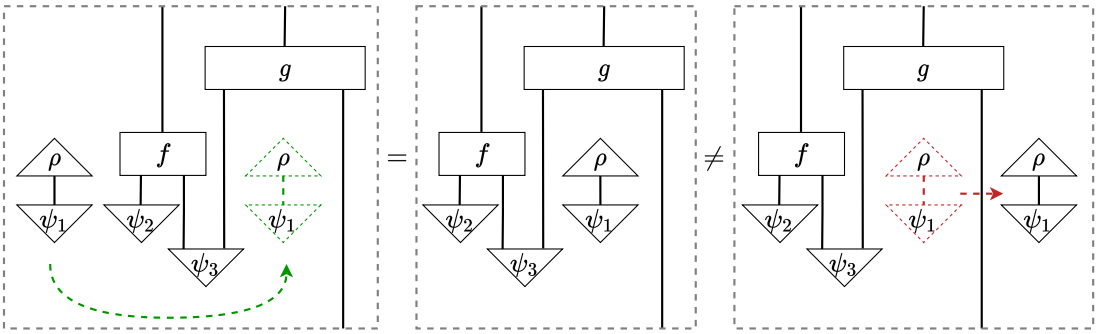
Monoidal categories give rise to special morphisms $\rho : I \to A$ from the unit object to any other object $A \in |\mathcal{C}|$. These morphisms are called states. As I can be considered the trivial object, every state corresponds to a unique initialisation of object A (Heunen & Vicary, 2018). This will be convenient later when we observe that in finite-dimensional Hilbert spaces, each state $\rho : I \to A$ corresponds to a unique vector in A. Morphisms $\psi : A \to I$ are called *effects*. States and effects have special representations in the graphical calculus, as the wire for the unit is not drawn. They respectively have no inputs and one output or one input and no outputs:

A state An effect

The graphical calculus is sound and complete for monoidal categories. It allows for easy derivations within a given category. We have:

Theorem 1 (Heunen and Vicary, 2018, Theorem 1.8)**.** *A well-typed equation between morphisms in a monoidal category follows from the axioms if and only if it holds in the graphical language up to planar isotopy.*

This means that two morphisms represented by their respective diagrams are equal if and only if we can continuously deform one diagram into the other. For example, we have:



(2.1)

We have to imagine each diagrammatic representation of a morphism surrounded by a box, which we cannot cross during deformations. The grey, dotted box indicates this. The first equality holds as we can move the $\psi_1 \circ \rho$ around to plane without crossing any wires to get it from the first position to the second position. This is indicated by the green, dotted arrow on the left. The second equality does not hold, as we cannot move the state-effect-pair to the third position on the right without crossing the right wire coming out of g. This is indicated by the red dotted arrow. Thus there is no planar isotopy. Here the true power of the diagrammatic representation becomes clearer. Writing out this (in-)equality in classical terms is much less readable. A proof of this (in-)equality becomes substantially more complicated.

2.3 Symmetric monoidal category

A *symmetric monoidal category* allows us to swap wires.

Definition 3 (Heunen and Vicary, 2018, Definition 1.17, 1.20)**.** *A* ***symmetric monoidal category*** $\mathcal{C}$ *is a monoidal category equipped with a natural isomorphism* $\sigma_{A,B} : A \otimes B \to B \otimes A$ *called the braid, which satisfies the hexagon equations*[2] *such that for all* $A, B, C \in |\mathcal{C}|$*, we have:*

$$\alpha_{B,C,A} \circ \sigma_{A,B\otimes C} \circ \alpha_{A,B,C} = (id_B \otimes \sigma_{A,C}) \circ \alpha_{B,A,C} \circ (\sigma_{A,B} \otimes id_C)$$

[2]The hexagon equations can also be written as commuting diagrams. We refer the interested reader to Heunen and Vicary (2018, Equation 1.15, 1.16).

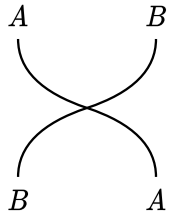
and:

$$\alpha^{-1}_{C,A,B} \circ \sigma_{A\otimes B,C} \circ \alpha^{-1}_{A,B,C} = (\sigma_{A,C} \otimes id_B) \circ \alpha^{-1}_{A,C,B} \circ (id_A \otimes \sigma_{B,C})$$

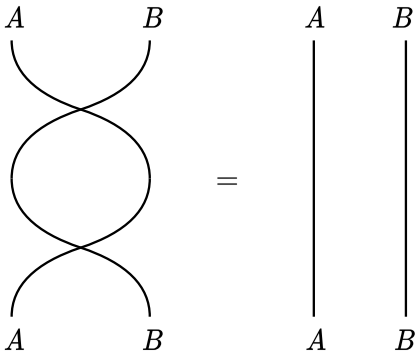
Additionally the braid is self inverse, i.e.:

$$\sigma_{B,A} \circ \sigma_{A,B} = id_{A\otimes B}$$

Graphically we represent the braid as swapping wires:

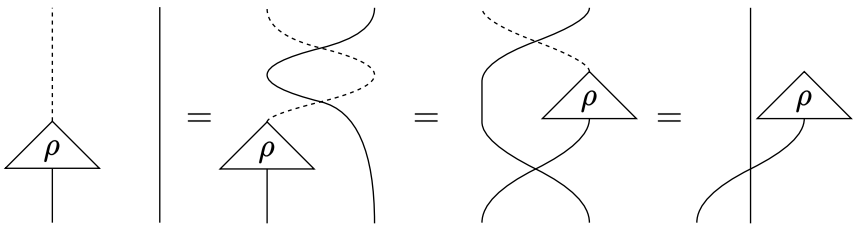


Thus the braid being its own inverse becomes:



Intuitively we simply straighten out the wire by pulling one over the other. Complex interactions can thus be simplified.

Symmetric, monoidal categories allow us to move states past wires. We have:



We draw the identity object as a dotted wire to illustrate our derivation. In the first step, we utilise the self inverse of the braid. Then we use the naturality of the braid, which allows us to move morphisms past the braid. In the final step, we straighten out the wires and omit the identity to make our final equation more

appealing. This ability to move states past wires will later become helpful in modelling conversational negation in sentences. It also means that the inequality in Equation 2.1 would be an equality in a symmetric, monoidal category. We apply this process to both the state and the effect to move the state-effect-pair past the wire.

2.4 Compact closed categories

The final category we are going to introduce are compact closed categories.

Definition 4 (Kartsaklis et al., 2016)**.** *A* ***compact closed category*** $\mathcal{C}$ *is a monoidal category where every object* $A \in |\mathcal{C}|$ *has a left and right adjoint* A^l *and* A^r *and morphisms:*

$$\epsilon^r_A : A \otimes A^r \to I \qquad \eta^r_A : I \to A^r \otimes A$$

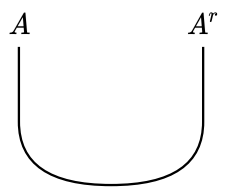
$$\epsilon^l_A : A^l \otimes A \to I \qquad \eta^l_A : I \to A \otimes A^l$$

such that the yanking equations hold:

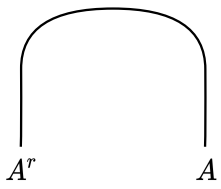
$$(id_A \otimes \epsilon^l_A) \circ (\eta^l_A \otimes id_A) = id_A \qquad (\epsilon^r_A \otimes id_A) \circ (id_A \otimes \eta^r_A) = id_A$$

$$(\epsilon^l_A \otimes id_{A^l}) \circ (id_{A^l} \otimes \eta^l_A) = id_{A^l} \qquad (id_{A^r} \otimes \epsilon^r_A) \circ (\eta^r_A \otimes id_{A^r}) = id_{A^r}$$

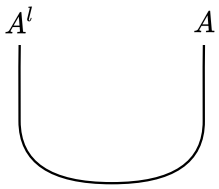
Diagrammatically we represent the morphisms as:



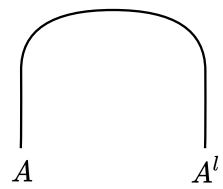
$\epsilon^r_A : A \otimes A^r \to I$



$\eta^r_A : I \to A^r \otimes A$

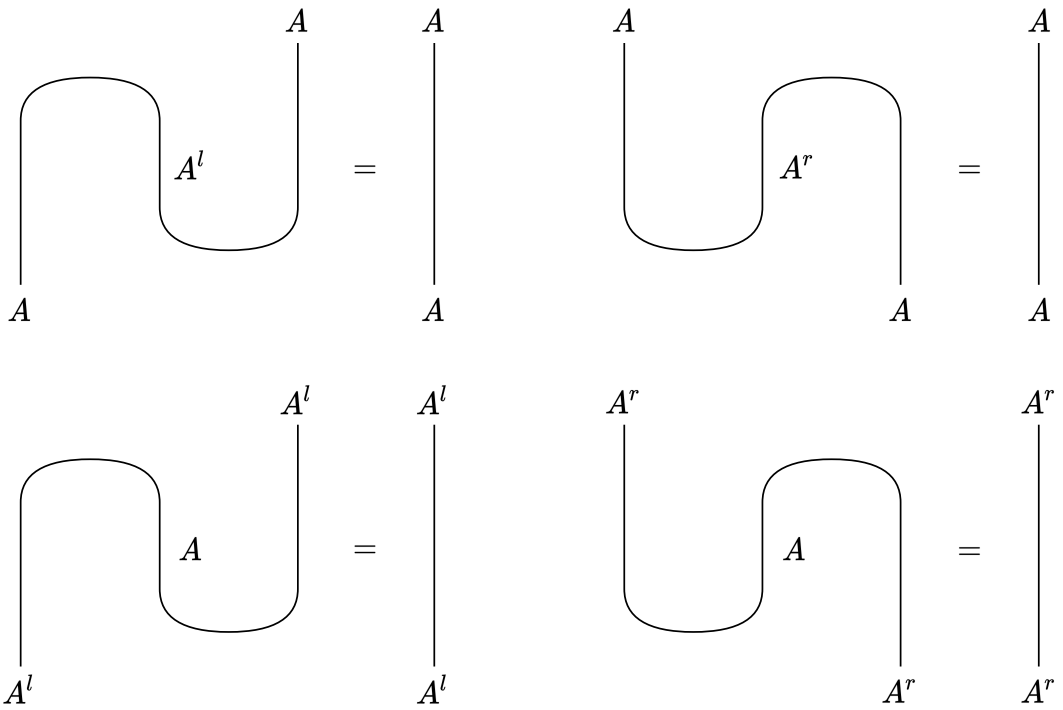


$\epsilon^l_A : A^l \otimes A \to I$



$\eta^l_A : I \to A \otimes A^l$

These morphisms are also referred to as *cup* and *cap*. The yanking equations correspond to:



Diagrammatically, these equations correspond to 'yanking' the wire straight.

When a compact closed category $\mathcal{C}$ is symmetric, we have $A^l = A^r =: A^*$. This single object is then called the dual of A. For the dual holds:

$$(A^*)^* = A$$

2.5 Monoidal functors

A functor is a mapping between two categories that preserves the categorical structure.

Definition 5. *Let $\mathcal{C}, \mathcal{D}$ be categories. A **functor** $F : \mathcal{C} \to \mathcal{D}$ is a mapping that:*

- *maps every object $X \in |\mathcal{C}|$ to an object $F(X) \in |\mathcal{D}|$*
- *maps every morphism $f : A \to B$ in $\mathcal{C}$ to a morphism $F(f) : F(A) \to F(B)$ in $\mathcal{D}$.*

The mapping of morphisms has to respect the identity and interact well with sequential composition, i.e.:

1. *for all* $X \in |\mathcal{C}|$, *we have* $F(id_X) = id_{F(X)}$

2. *for all morphisms* $f : A \to B, g : B \to C$, *we have* $F(g \circ f) = F(g) \circ F(f)$

Among monoidal categories, the stricter *monoidal functors* are defined to preserve the monoidal structure. We have:

Definition 6 (Kartsaklis et al., 2016)**.** *Let* $\mathcal{C}, \mathcal{D}$ *be monoidal categories. A* ***monoidal functor*** $F : \mathcal{C} \to \mathcal{D}$ *is a functor that respects the monoidal unit and tensor product. That is:*

1. *there exists a morphism*[3] $I_\mathcal{D} \to F(I_\mathcal{C})$ *in* $\mathcal{D}$

2. $F(A) \otimes F(B) \to F(A \otimes B)$ *is a natural transformation, which satisfies the corresponding coherence conditions*

Such a functor is called ***strongly monoidal*** *when the morphism and natural transformation are invertible.*

[3]We use subscripts to the monoidal units to make explicit the category to which they belong. In particular, this morphism is in $\mathcal{D}$ from the monoidal unit of $\mathcal{D}$ to the image of the monoidal unit of $\mathcal{C}$ under F.

Modelling language

3

Compositional distributional semantics

Contents

Historically, linguists differentiate between syntax — grammatical rules — and semantics – the meaning of words and sentences. For both, computer linguists have devised computational methods to analyse their respective aspects.

Lambek's (1999) pregroup grammar can be utilised to check whether sentences are typed correctly, i.e. whether they are grammatically correct. However, while pregroup grammar can model the interaction between words to form coherent sentences, it does not attribute meaning to the individual words. It, therefore, cannot provide meaning for the entire sentence.

Distributional semantics, based on the distributional hypothesis (Firth, 1957), provides a way to model the meaning of a word w as vectors calculated from a large text corpus, for example, by considering the words with which w often co-occurs. However, distributional semantics struggles with larger sentences.

One of the main differences between words and sentences is that to understand a word, we need to know its meaning. New words are impossible to understand without an explanation or sufficient context. To understand a sentence, it is sufficient to know the meaning of its parts and the way they interact. The sentence itself becomes a composition of its smaller parts. Therefore humans can easily understand sentences that they have never encountered before.

To model the meaning of sentences, Coecke et al. (2010) combine grammar with the meaning of words to derive the meaning of sentences. They observe that both the representation of grammar provided by Lambek (1999) and vector spaces used to model the meaning of words have the same categorical, compact closed structure. Thus, they propose the DisCoCat framework. DisCoCat provides a method to derive meanings of sentences from their grammatical structure and the meaning of their words. Later, Coecke (2020) generalises the framework to DisCoCirc by observing that sentences are not fixed meaning states but processes that update the already existing notions we have in our heads. This observation allows Coecke to compose sentences to model the meaning of entire texts.

These observations were made possible by the categorical abstraction for both meaning and grammar. The diagrammatic language, applied in this process, was originally introduced for quantum computing (Abramsky & Coecke, 2004). The extensive use of the tensor product makes this approach to natural language processing exponentially expensive on classical computers. For that reason, recent research has focused on implementing these frameworks on quantum computers (Coecke et al., 2020; Lorenz et al., 2021).

In this section, we will introduce a method to represent meaning with finite-dimensional vectors. Next, we will present Pregroup grammar, a model for grammar in language. We will then introduce the DisCoCat framework, which we generalise

to DisCoCirc. Both DisCoCat and DisCoCirc were initially introduced with vectors and pregroup grammar. However, they are open to any other representation of meaning and grammar, as long as they have the same categorical structure. We will introduce the frameworks with these examples due to their simplicity. In particular, with the move to DisCoCirc, the choice of grammar will become less relevant. A lot of the complexity required to express language in a 1-dimensional string (i.e. written text or spoken language) will be stripped away. Finally, we will define positive operators, an extension of vectors, as a richer representation of meaning. We will later utilise positive operators in the experimental validation of our proposed conversational negation framework (Section 4.5).

3.1 Representing words

3.1.1 Distributional semantics

Distributional semantics is built around the distributional hypothesis. This hypothesis, as popularized by Firth (1957), asserts that:

> a word is characterised by the company it keeps

This hypothesis allows us to represent words and their meaning as a simple vector in a finite-dimensional vector space. The meaning of all words can, for example, be computed from a large corpus of texts via co-occurrence. To calculate the meaning of words, first n context words $c_1, ..., c_n$ are being selected. The meaning of some word w is then represented as an n-dimensional vector. Each entry i in the vector is calculated by counting how often w co-occurs with the context words c_i, i.e. how often w and c_i are used in close proximity to one another.

Thus for example, if the context words contain *sweet, sour, green* and *blue*, then the word `apple` could have values 4, 3, 3, 0 at the respective entries in it's meaning vector. This reflects that apples are often said to be sweet or sour and green but are never mentioned in the company of blue.

While this might seem like a primitive approach to word meaning, it gives remarkable results. Coecke et al. (2010) give a good overview of interesting applications of

distributional semantics, including word sense discrimination and disambiguation (McCarthy et al., 2004; Schütze, 1998), language modeling (Bellegarda, 2000) and document retrieval (Salton et al., 1975).

Many of these results are achieved by modelling the vectors in Hilbert spaces, which are vector spaces with an inner product. This inner product provides a distance function to measure how related two words are. Words that are closely related tend to live close to each other in the respective Hilbert space. Additionally, Hilbert spaces provide the necessary structure for our representation of meaning to form a compact closed category.

While distributional semantics gives good results for modelling words in isolation, it is inadequate for phrases or sentences. This is in part because the longer phrases get, the rarer they are in any given text. Therefore a substantially larger text corpus is required to gather comparable amounts of data. Additionally, distributional semantics can only model phrases already present in the corpus and cannot derive the meaning of new sentences.

Some approaches to composing word meanings include the *bag-of-words-view* where meaning vectors are simply averaged, ignoring their grammatical interaction. Thus sentences like "`Humans eat chicken.`" and "`Chicken eat humans.`" give identical result. These methods perform surprisingly well, for example, in automatic essay grading (Landauer et al., 1997) and coherence assessment (Foltz et al., 1998). Mitchell and Lapata (2010) provide alternative element-wise combination methods. Many of these either ignore grammatical structure or struggle with comparing grammatically different sentences. Reasons for composing meanings and some approaches are outlined in Baroni (2013).

3.1.2 Hilbert spaces as a compact closed category

Finite-dimensional Hilbert spaces form a category **FHilb** with objects being the finite-dimensional Hilbert spaces and morphisms being linear maps between Hilbert spaces. This category is symmetric and compact closed, with the monoidal product being the tensor product (Coecke & Paquette, 2011). As this category is symmetric,

for some object V, we have $V^l = V^r$. We refer to this single object as the dual V^*. Every Hilbert space V is isomorphic to its dual V^*. As we consider strict monoidal categories, we can therefore treat the dual map as the identity. Thus we have $V = V^*$. In terms, η and ϵ are defined as follows:

$$\eta_V : I \to V \otimes V :: 1 \mapsto \sum_i \vec{n}_i \otimes \vec{n}_i$$
$$\epsilon_V : V \otimes V \to I :: \sum_{i,j} c_{i,j} \vec{v}_i \otimes \vec{w}_j \mapsto \sum_{i,j} c_{i,j} \langle \vec{v}_i | \vec{w}_i \rangle$$

for some Hilbert space V with some basis $\{\vec{n}_i\}_i$. For the second equation, we utilise the bra-ket notation, where $\sum_{i,j} c_{i,j} \vec{v}_i \otimes \vec{w}_j$ is some vector in $V \otimes V$. The yanking equations can then be verified. For example, for the first equation we have:
Let $\vec{v} \in A$ be a vector. Then:

$$\begin{aligned}((id_A \otimes \epsilon^l_A) \circ (\eta^l_A \otimes id_A))(\vec{v}) &= (id_A \otimes \epsilon^l_A)((\sum_i \vec{n}_i \otimes \vec{n}_i) \otimes \vec{v}) \\ &= \sum_i \vec{n}_i \otimes \langle \vec{n}_i | \vec{v} \rangle \\ &= \vec{v} \\ &= id_A(\vec{v})\end{aligned}$$

Thus $((id_A \otimes \epsilon^l_A) \circ (\eta^l_A \otimes id_A))(\vec{v}) = id_A(\vec{v})$ for all $\vec{v} \in A$. Therefore we can conclude that the two maps are equivalent.

3.2 Representing grammar

3.2.1 Pregroup grammar

Pregroup grammar (Lambek, 1999) provides a mathematical framework to analyse the structure of natural language. In pregroup grammar, each class of words, such as nouns or transitive verbs, is assigned a type. If the words in a sentence combine to reduce to some specific type, we consider the sentence typed well, i.e. grammatically correct. On the other hand, if the types of the words do not reduce to this specific type, the sentence is ill typed, i.e. grammatically incorrect.

First, we define:

Definition 7 (Coecke et al., 2010)**.** *A **partially ordered monoid** $(P, \leq, \cdot, 1)$ is a partially ordered set $(P, \leq)$, equipped with a monoid multiplication $- \cdot -$ and an object 1. Where the monoid multiplication is associative and 1 is the unit, i.e.:*

$$\begin{aligned} \forall p, q, r \in P, \quad (p \cdot q) \cdot r &= p \cdot (q \cdot r) \\ \forall p \in P, \quad p \cdot 1 = p &= 1 \cdot p \end{aligned}$$

such that:

$$\forall p, q, r \in P, \quad p \leq q \implies p \cdot r \leq q \cdot r \quad \wedge \quad r \cdot p \leq r \cdot q$$

Based on Definition 7, we can define:

Definition 8 (Coecke et al., 2010)**.** *A **pregroup** $(P, \leq, \cdot, 1, (-)^l, (-)^r)$ is a partially ordered monoid $(P, \leq, \cdot, 1)$ equipped with the unitary operations $(-)^l$, $(-)^r$ called left and right adjoint, such that:*

$$\forall p \in P, \quad p^l \cdot p \leq 1 \leq p \cdot p^l \quad \wedge \quad p \cdot p^r \leq 1 \leq p^r \cdot p$$

We normally omit the dot when composing elements. Thus instead of $p \cdot q$ we write pq.

A pregroup grammar can be freely generated over a set of basic elements (Kartsaklis et al., 2013). A simple example, given in Coecke et al. (2010), is built on the two elements s and n. The objects of the pregroup grammar are thus n, s and all the infinitely many adjoints we can construct inluding n^r, n^l and $(n^l)^l = n^{ll}$. Each word is then assigned a type made up of compounds of the elements in our grammar. We then say that a sentence is typed correctly if the types of its components can be reduced to s. We assign nouns to the type n and transitive verbs to the type $n^r s n^l$. This reflects that a transitive verb expects a noun on the left and on the right to form a grammatically correct sentence. The sentence "`Alice loves Bob`" will then become:

`Alice`	`loves`	`Bob`
noun	transitive verb	noun
n	$n^r s n^l$	n

Therefore, the type of the sentence is $nn^r sn^l n$, which, using the rules of the pregroup, we can reduce to:

$$nn^r sn^l n \leq 1sn^l n \tag{3.1}$$

$$\leq 1s1 \tag{3.2}$$

$$= s \tag{3.3}$$

Thus this sentence can be reduced to s and is therefore correctly typed.

We observe a couple of caveats. First of all, this approach only considers the grammatical role of words. Therefore the sentence "`Travel loves house.`", which is also of the form *noun – transitive verb – noun* is considered grammatically correct. Secondly, we have to admit that this is a massively simplified example. Pregroup grammar becomes a lot more complex when trying to model all words in the English language. For example, some words might be able to play different roles depending on how they are used; the word `like` could be used as a verb ("`I like you.`") or as a preposition ("`It is like this.`"). Various approaches have been developed to overcome these challenges, such as probabilistic approaches (Kornai, 2011) or other grammatical systems with the same categorical structure (Yeung & Kartsaklis, 2021).

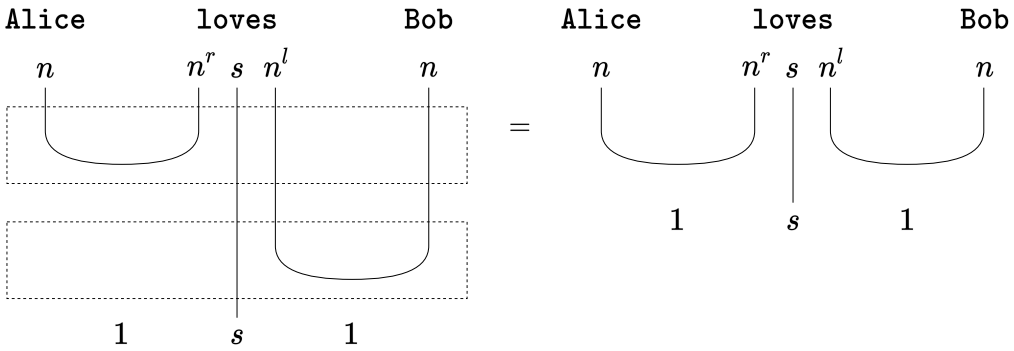
3.2.2 Pregroups as a compact closed category

Pregroups form a compact closed category (Coecke et al., 2010). The objects of the category are the types. Any two types a, b are connected by a morphism $a \to b$ if and only if $a \leq b$. This category is monoidal, where $- \cdot -$ provides the monoidal product. The monoidal unit is then given by 1. The cup and cap, required for the compact closure, are respectively:

$$\epsilon^r_A : A \cdot A^r \leq 1 \qquad \eta^r_A : 1 \leq A^r \cdot A$$

$$\epsilon^l_A : A^l \cdot A \leq 1 \qquad \eta^l_A : 1 \leq A \cdot A^l$$

Thus we can visualise Equation 3.1 diagrammatically as:



On the left-hand side, the first dotted box corresponds to the reduction to Equation 3.1 and the second dotted box corresponds to the second reduction to Equation 3.2. We can rewrite the two reductions to a single reduction on the right-hand side.

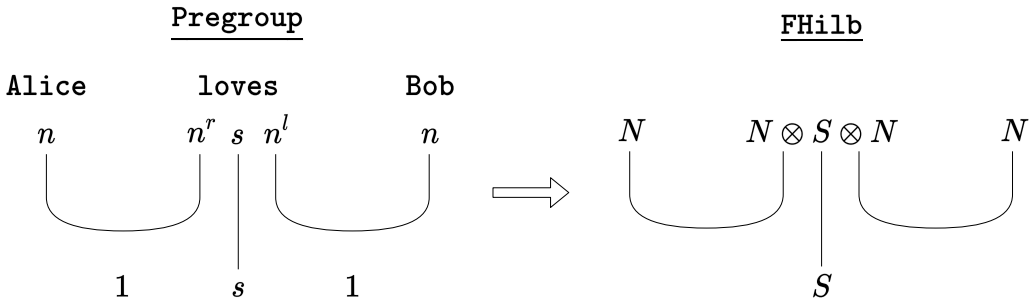
3.3 DisCoCat - from words to sentences

Coecke et al. (2010) propose a categorical, compositional, distributional approach to natural language processing. They observe that pregroup grammars and vector spaces form compact closed categories. The idea is that grammar informs the interaction between the words. This interaction is then mapped to the vectors, which represent the meanings. Through this interaction, the vectors are combined to form the meaning of the entire sentence. We thus get the meaning of a sentence from the meaning of the words interacting informed by grammar. We use **cat**egory theory to **co**mpose **dis**tributional meanings (DisCoCat).

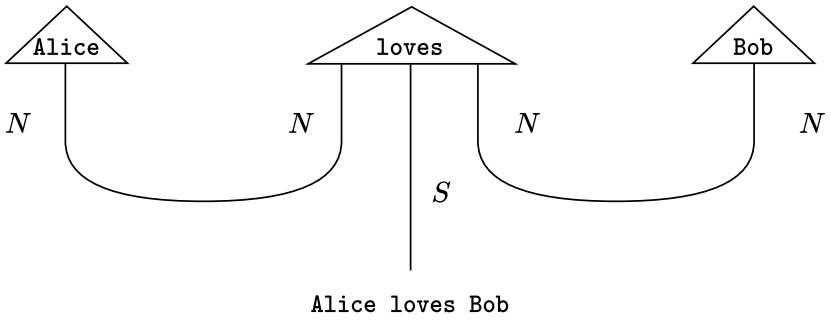
To do this, we need a strong monoidal functor from the grammar category to the meaning category. This functor then maps objects to objects and morphisms to morphisms. Thus for each object in the pregroup grammar, we have to assign a corresponding finite-dimensional vector space to represent the meaning of the words. We can map the noun object n to some vector space N of dimension d. With this choice, we decide that all nouns will be represented by a vector of d dimensions. This choice, therefore, informs the amount of information we can convey per word. It is a design choice that will have to be made depending on the respective task. Having mapped n to N, we map the adjoints of n to the adjoint of N. For example,

we have n^l maps to N^l. However, as finite vector spaces are self-dual, we have n maps to $N^l = N$. Thus all adjoints of n map to N. Similarly, we map the sentence object s and its adjoints to some vector space S. Once again, the dimension of S determines the size of the vectors representing sentences. The individual words then live in compounds of these vector spaces, according to the type they have been assigned (nouns in N and transitive verbs in $N \otimes S \otimes N$).

Once the choice of vector spaces has been made, the rest of the functor follows trivially. For example, the cups and caps of n in the pregroup will be mapped to the cups and caps of N in the vector space. Diagrammatically this can be represented as:



where the arrow is the monoidal functor. While this looks rather trivial, it is a very powerful observation. The cups on the left simply correspond to the $\leq$ relation in our pregroup grammar. They have no other meaning. The cups on the right, in the category **FHilb**, are operations on vectors that describe how to derive a vector in S from three vectors in N, $N \otimes S \otimes N$ and N respectively. Therefore on the right, we have an operation on meaning. Observing that vectors in some vector space A have a one-to-one correspondence with states $I \to A$, we can now plug in the meaning of our concrete words `Alice, love` and `Bob` to calculate the meaning of the entire sentence.



Thus the overall diagram corresponds to one big state `Alice loves Bob` : $I \to S$; it has no inputs and one output in S. This state corresponds to a vector in S, which represents the meaning of our sentence. While our diagram looks to represent a morphism $I \otimes I \otimes I \to S$, due to the unitality of the object I this is equal to a morphism $I \to S$. In a sense, we could imagine one large triangle around the entire sentence, giving us a state with no inputs and one output in S.

In contrast to some previous approaches to combining vectors, the DisCoCat framework guarantees all sentences to live in the same space, independent of their grammatical structure. Amongst other things, this allows the comparison of grammatically different sentences. This is something other compositional approaches struggle with (Coecke et al., 2010). Implementations of the DisCoCat outperform other frameworks in certain academic benchmarks (Grefenstette & Sadrzadeh, 2011; Kartsaklis & Sadrzadeh, 2013).

More formally, we can define:

Definition 9 (Sadrzadeh et al., 2018)**.** *An instantiation of the DisCoCat framework is given by a quantuple* $(\mathcal{C}_{syn}, \mathcal{C}_{sem}, F, [\![]\!])$ *where* $\mathcal{C}_{syn}$ *is a compact close category for the syntax.* $\mathcal{C}_{sem}$ *is a compact closed category for the semantics.* $F : \mathcal{C}_{syn} \to \mathcal{C}_{sem}$ *is a strongly monoidal functor from the syntax category to the semantics category.* $[\![]\!] : \Sigma^* \to C_{sem}$ *is a map from any string in the (English) language to the category of semantics.*

The meaning of a string of words $w_1, ..., w_n$ *can then be calculated as*

$$[\![w_1 ... w_n]\!] \coloneqq F(\alpha)([\![w_1]\!] \otimes ... \otimes [\![w_n]\!])$$

where the morphism α *in the syntax category denotes the grammatical structure of the string* $w_1 ... w_n$*.*

While we presented the framework with pregroups and vectors, it is open to any choice of grammar and meaning as long as they have the same categorical structure. Other grammars that have been explored include CCG (Yeung & Kartsaklis, 2021) while other meaning representations include positive operators

(Coecke & Meichanetzidis, 2020), Montague-style Boolean-valued semantics (Coecke et al., 2010) and conceptual spaces (Bolt et al., 2019; Tull, 2021). Therefore, we do not exclusively rely on the distributional hypothesis as long as we find a suitable representation for meanings. For example, recent implementations on actual quantum computers learned the meanings of words as part of a training process, thereby mixing standard supervised learning with this new framework (Lorenz et al., 2021).

3.4 DisCoCirc - from sentences to texts

The DisCoCat framework provides a method to combine the meaning of words to the meaning of sentences. The DisCoCirc framework (Coecke, 2020) combines the meaning of sentences to the meaning of entire texts. The central observation for this transition is that:

A sentence is not a state, but a process

The underlying idea is that a sentence is not something that has a fixed meaning, i.e. a (meaning) state. Instead, we have preconceived meanings in our heads, such as concepts or people. Sentences update these meanings. They become processes that take our existing, preconceived meanings and provide additional information, with which we can update them.

Coecke differentiates between:

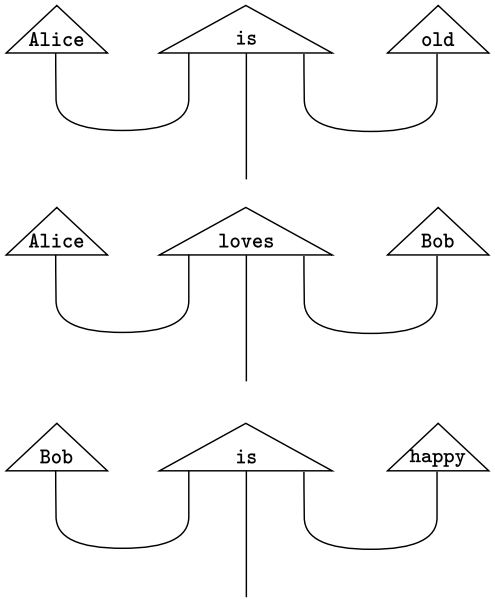
- *static words*: words which are not altered by the text
- *dynamic words*: words which are altered by the text

For example, the actors in a story are represented by dynamic words; we learn who they are, how they interact and what they do.

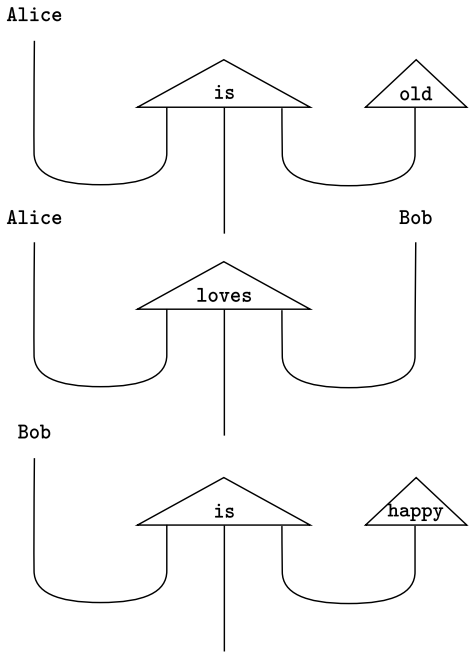
We can consider the following short text:

```
Alice is old.

Alice loves Bob.

Bob is happy.
```

The three sentences in the DisCoCat framework look like this:



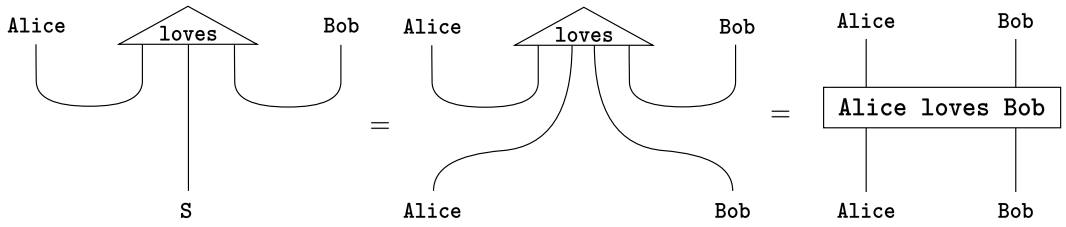
But we can observe that throughout this text, we gain increasing information about `Alice` and `Bob`. We can consider them as dynamic words. Instead of closing their wires, we open them up as such:



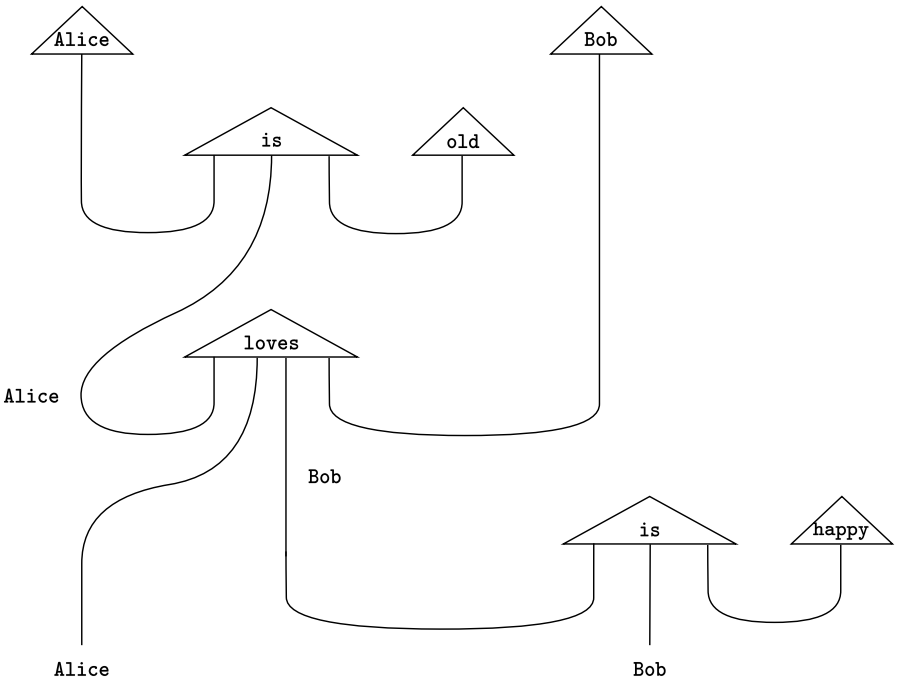
Therefore, the dynamic words are not states but meanings that are already present and carried on wires. We thus assume that there exists a wire, labelled `Alice` and `Bob` which carries our preconceived notion of these two dynamic words.

While initially, sentences had no inputs and one output, they now have inputs and outputs. The sentences become processes that take in the meaning of the dynamic words and produce a sentence meaning.

One restriction of DisCoCat is that it does not give any indication of the meaning representation for sentences. In the move to DisCoCirc, we impose that the type of a sentence is the tensor product of the dynamic words. In a sense, sentences update the meaning of the dynamic words. We, for example, have:

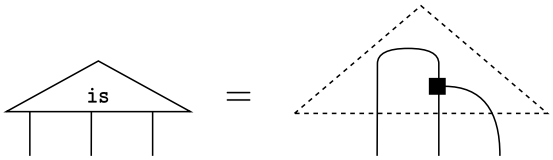


Therefore the sentence becomes a morphism that takes in two wires and outputs two wires. The right-hand side shows this high-level view; the sentence becomes a process that updates the dynamic meanings of `Alice` and `Bob`. This allows us to sequentially compose the processes to one big circuit in which we initialise both `Alice` and `Bob` only once in the very beginning.

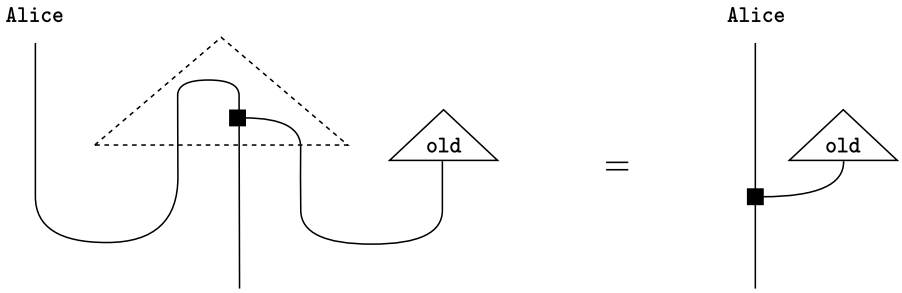


The original `Alice` and `Bob` states contain all preconceptions we have associated with their names — such as gender or origin. However, throughout the text, we increasingly get new information such that at the end of the text, we have new values on the corresponding wires.

Next, we observe that some words do not carry a meaning but rather an operation. The verb `to be` can appear in many, completely different contexts. Rather than bringing new meaning, it informs how the surrounding words interact. We can model the word `is` as such:



where the black box is an update mechanism (Coecke, 2020). In Section 3.5.5 we propose some concrete update mechanisms for our choice of meaning representation. The first sentence thus looks like this:

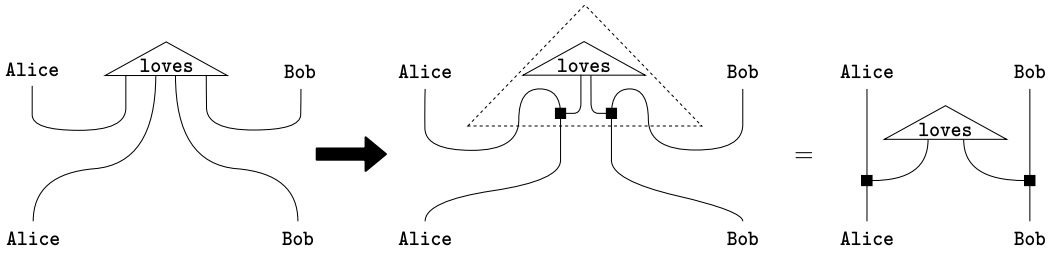


where we use the yanking equations to deform the diagram to get the right-hand side. The sentence can be seen to updating `Alice` with the property `old`.

Similarly, entire classes of words have an operational meaning rather than an inherent meaning. This includes relative pronouns (Sadrzadeh et al., 2013, 2014) and conjunctions (Duneau, 2021). In this thesis, we will similarly propose an operation for negation.

Going a step further, Coecke and Wang (2021) observe that language is highly complex but has to be compressed into one-dimensional strings to make it usable for humans. Trying to remove the arbitrary, unnecessary complexity imposed by human limitations, they propose introducing internal wiring for all grammatical types, even

those that additionally carry meaning. The internal wiring is manually imposed based on our understanding of human grammar. Therefore, in the diagrammatic representation, we capture not only the meaning of the words but also their grammatical role. When combining the grammatical role with the grammatical interaction, the latter being informed by the grammar, language circuits simplify considerably. They propose the idea of a grammatical normal form, which allows a general simplification of all language diagrams. In this view, even the transitive verb can be seen as an update to the evolving meanings, using the verb form introduced in Grefenstette and Sadrzadeh (2011) and Kartsaklis and Sadrzadeh (2014). For the example "`Alice loves Bob`", we have:



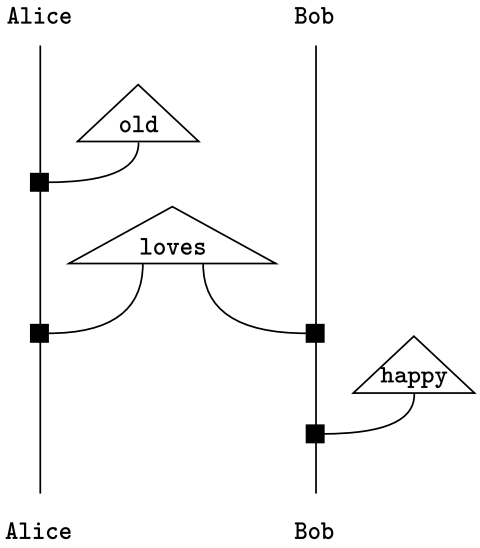
This shows that the word `love` can simply be seen as an update that connects both `Alice` and `Bob`.

These simplifications through the introduction of internal wires ensure that sentences with the same informational content will act similarly. For example, the two sentences "`Alice likes the flowers that Bob gives Claire`" and "`Bob gives Claire the flowers that Alice likes`" might convey the same informational content but have very different grammatical structures. Therefore, in classical approaches, they often get very different representations. Through the internal wires, these differences get stripped away to show their similarity while preserving the subtle, causal differences[1](Coecke & Wang, 2021).

In this thesis, we will assume that all sentences can be modelled as a collection of meaning states that inform the evolving meanings via updates. Each meaning

[1]Does `Bob` give `Claire` the `flowers` because `Alice` likes them? Or does `Alice` happen to like the `flowers` that `Bob` picked?

state updates one or multiple wires. Furthermore, we will assume that each word acts upon the wires in consecutive order. This is an oversimplification that does not consider more complex words. However, it suffices to convey the ideas proposed in this thesis. Future work will have to generalise this to all language circuits once they have been fully formalised[2]. Thus in this view, our example text looks like this:



3.5 A meaning representation - positive operators

3.5.1 Definition

DisCoCat and DisCoCirc allow for any compact closed category to be utilised to represent meaning. We introduced the frameworks at the example of the category of finite-dimensional Hilbert spaces and linear maps **FHilb**. Throughout the remainder of this thesis, we will use the richer category of positive operators with completely positive maps, known as CPM(**FHilb**). The objects of CPM(**FHilb**) are defined as:

Definition 10. *A **positive operator** is a complex matrix that is:*

- *Hermitian - equal to its own conjugate transpose*
- *positive semidefinite - has only non-negative eigenvalues*

[2]This work is currently still in progress.

The morphisms of CPM(**FHilb**) are:

Definition 11. *A **completely positive map** is a linear map that preserves the positivity of operators, i.e. maps positive operators to positive operators.*

The category CPM(**FHilb**) can be seen as an extension to the category of finite-dimensional Hilbert spaces **FHilb**. It can be obtained from **FHilb** via the CPM construction, which was originally introduced by Selinger (2007). In this construction, pure state positive operators are constructed from a unit vector $|v\rangle$ of a finite-dimensional Hilbert space by taking the outer product $|v\rangle \langle v|$. All other positive operators are a linear combination of pure states. This CPM construction has a beautiful, intuitive diagrammatic representation found in Selinger (2007).

CPM(**FHilb**) offers two advantages over **FHilb**, due to which this category has been chosen. First, in contrast to vectors, which by themselves cannot model an ordering on meaning (Balkir et al., 2016), positive operators have ordering relations, which allow us to measure entailment (Section 3.5.3). Secondly, positive operators can be used to encode ambiguity and allow for later disambiguation (Coecke & Meichanetzidis, 2020; Piedeleu et al., 2015) (Section 3.5.4).

3.5.2 Kernel and support

Throughout this thesis, we will be talking about the *kernel* and *support* of positive operators. The kernel and support of a matrix are defined in linear algebra. As positive operators are matrices, these definitions apply. We will recall the definitions and elaborate on what they mean for positive operators.

Definition 12. *Let M be a matrix with corresponding linear transformation $T : \mathcal{C}^n \to \mathcal{C}^m$. The **kernel** of M, written $ker(M)$, is the set of vectors $\vec{v} \in \mathcal{C}^n$ such that $T(\vec{v}) = \vec{0}$. It forms a subspace of $\mathcal{C}^n$.*

The support is then defined as all other vectors, i.e.:

Definition 13. *Let M be a matrix with corresponding linear transformation $T : \mathcal{C}^n \to \mathcal{C}^m$. The **support** of M, written $supp(M)$, is the set of vectors $\vec{v} \in \mathcal{C}^n$ such that $T(\vec{v}) \neq \vec{0}$.*

Any positive operator A with spectral decomposition $\sum_i \lambda_i |i\rangle \langle i|$ has eigenvectors $|i\rangle$ and corresponding eigenvalues λ_i. By the definition of positive operators we have $\lambda_i \geq 0$ for all i. The kernel is then spanned by the eigenvectors with eigenvalue 0.

When representing words as positive operators, we can view the eigenbasis as the properties of the word. The eigenvectors spanning the kernel are not present in the word, while the eigenvectors in the support are present with degree λ_i.

3.5.3 Entailment

A word w_1 entails a word w_2 if every w_1 is a w_2. Thus `dog` entails `animal` as every `dog` is an `animal`. Bankova et al. (2019) and Lewis (2019) propose some measures to quantify entailment between positive operators. Some desirable properties for entailment measures are:

- They are graded - most `dogs` are `pets`. Thus `dog` highly (but not fully) entails `pet`
- They are asymmetric - all `dogs` are `animals` but not all `animals` are `dogs`. Thus `dog` fully entails `animal` but not the other way around.
- They are pseudo-transitive - `dogs` are `animals` and `dogs` are `fluffy` thus `animals` can be `fluffy`.

One order on positive operators is the Löwner order:

Definition 14. *The **Löwner order**, $\sqsubseteq$, is a partial order on positive operators. Let A, B be positive operators. We have:*

$$A \sqsubseteq B \quad \Leftrightarrow \quad B - A \quad \textit{is a positive operator}$$

This corresponds to saying

$$\mathsf{A} \sqsubseteq \mathsf{B} \quad \Leftrightarrow \quad \exists \text{ a positive operator } \mathsf{D} \text{ such that } \mathsf{A} + \mathsf{D} = \mathsf{B}$$

While this order is asymmetric, it is not graded.

We will consider three generalisations on the Löwner order as graded entailment measures. In our later experiments, we will evaluate all three of them.

Bankova et al. (2019) propose a weighted generalisation of the Löwner order, called k-hyponomy.

Definition 15. ***k-hyponomy****, in short k_{hyp}, is a weighted partial order on positive operators. Let A, B be positive operators. We have:*

$$A \sqsubseteq_k B \quad \Leftrightarrow \quad B - kA \quad \textit{is a positive operator}$$

for some maximal $k \in [0, 1]$. We then say A entails B with strength k.

Lewis (2020) observes that this corresponds to saying there exist positive operators D and E with E of the form $\mathsf{E} = (1 - k)\mathsf{A}$ such that:

$$\mathsf{A} + \mathsf{D} = \mathsf{B} + \mathsf{E}$$

She points out that the *k-hyponomy* is not robust to errors, as it requires the support of A to be fully included in the support of B for a non-zero entailment. We can generalise k_{hyp} by utilising Bankova et al. (2019, Theorem 2), which states

Theorem 2 (Bankova et al., 2019, Theorem 2)**.** *For positive operators A, B such that:*

$$supp(A) \subseteq supp(B)$$

the maximum k such that $B - kA$ is a positive operator is given by $k = \frac{1}{\lambda}$ where λ is the maximum eigenvalue of $B^{+}A$. Where B^{+} is the Moore-Penrose inverse, which we will later introduce as the support-inverse-negation (Section 4.2.1.1).

We can then generalise k_{hyp} by lifting the restriction that the support of A has to be contained in the support of B. We can calculate the generalised k_{hyp} as $\frac{1}{\lambda}$ in all cases. This means that $\mathsf{B} - k\mathsf{A}$ does not have to result in a positive operator, which is undesirable. However, as a plausibility measure, this is tolerable. We note that in our experiments (Section 4.5) this generalisation performs considerably better than the normal k_{hyp} .

Lewis (2019) instead proposes to lighten the restriction on the error term E by defining it to be a positive operator constructed by diagonalising $\mathsf{B} - \mathsf{A}$ and setting all positive eigenvalues to 0 and taking the absolute value of all negative eigenvalues (Lewis, 2019). She then defines the two more robust measures k_{E} and k_{BA}.

Definition 16. *Let A, B be positive operators then:*

$$k_E = 1 - \frac{\|E\|}{\|A\|}$$
$$k_{BA} = \frac{\sum_i \lambda_i}{\sum_i |\lambda_i|} = \frac{Trace(D - E)}{Trace(D + E)}$$

where λ_i are the eigenvalues of E and $\|-\|$ is the Frobenious norm.

k_{E} is an asymmetric measure that ranges from 0 (when $\mathsf{E} = \mathsf{A}$) to 1 (when $\mathsf{E} = 0$). k_{BA} is symmetric up to a factor of -1 in that $k_{\mathsf{BA}}(\mathsf{A}, \mathsf{B}) = -k_{\mathsf{BA}}(\mathsf{B}, \mathsf{A})$. It ranges from -1 (when $\mathsf{D} = 0$) to 1 (when $\mathsf{E} = 0$). We will explore all three entailment measures experimentally (Section 4.5).

3.5.4 Ambiguity

A meaning is ambiguous if it can be interpreted in more than one way. A simple example could be the word `capital` which might refer to money or a city. Positive operators allow encoding multiple meanings at once via mixing - taking weighted sums over the different meanings. We could thus define the positive operator of `capital` as:

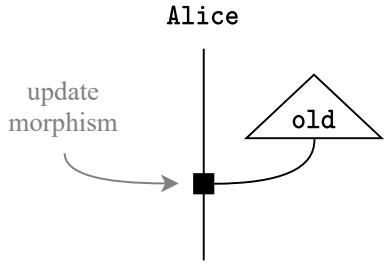
$$[\![\texttt{capital}]\!] = [\![\texttt{capital the money}]\!] + [\![\texttt{capital the city}]\!]$$

where we use the double brackets to indicate that we refer to the positive operator of a particular word.

Coecke and Meichanetzidis (2020) show how different meaning updates can be used to disambiguate ambiguous meanings. The encoding of ambiguity will be relevant when modelling negation, the interpretation of which can be ambiguous depending on the context.

3.5.5 Composition operations

An essential aspect of the DisCoCirc framework is the process of updating meaning. For example, the representation for "`Alice is old`" updates the dynamic meaning of `Alice` with the word `old`. Diagrammatically we have:



The black box is the update morphism, which takes in the two meanings and produces a new one.

Positive operators allow for various update operations, over which De las Cuevas et al. (2020) provides a good overview. We will consider four of those; the *spider, fuzz, phaser* and *diag*. The other options are undesirable, discarding too much information, as pointed out by De las Cuevas et al. (2020). While some of these operations have psychological motivation, the most optimal choice will be established empirically during our experiments (Section 4.5).

3.5.5.1 Spider

Definition 17. *Let A, B be positive operators, then*

$$spider(A, B) \coloneqq U_s(A \otimes B)U_s^\dagger$$

where

$$U_s = \sum_i |i\rangle \langle ii|$$

where $\{|i\rangle\}_i$ is some basis.

The spider forms a dagger special commutative Frobenius algebra (Coecke & Kissinger, 2017). The commutativity is an undesired property (Coecke & Meichanetzidis, 2020), making the order of sentences irrelevant. With such an update operation, the following two excerpts from recipes would be considered identical (Coecke & Meichanetzidis, 2020):

Recipe 1	Recipe 2
`Add eggs to the mixture.`	`Bake the mixture.`
`Bake the mixture.`	`Add eggs to the mixture.`

We observe that the spider is basis dependent. Thus to use the spider, one first has to choose a basis, which will determine the basis of the output.

3.5.5.2 Fuzz

Definition 18. *Let* A, B *be positive operators with* $B = \sum_i x_i P_i$*, then*

$$fuzz(A,\ B) \coloneqq \sum_i x_i P_i \circ A \circ P_i$$

Lewis (2020) calls this operation Kmult. Intuitively, the fuzz does a fuzzy update on the positive operator. It returns a mixture of having done multiple different updates, one for each projector P_i (Coecke & Meichanetzidis, 2020).

3.5.5.3 Phaser

Definition 19. *Let* A, B *be positive operators with* $B = \sum_i x_i^2 P_i$*, then*

$$phaser(A,\ B) \coloneqq (\sum_i x_i P_i) \circ A \circ (\sum_i x_i P_i)$$

Lewis (2020) calls this operation Bmult. Van de Wetering (2018) shows that it corresponds to the quantum Baysian update.

This operation updates A with the square root of B. While this may seem unintuitive at first, Coecke and Meichanetzidis (2020) show that this operation generalises the spider operation. Furthermore, we note that both the fuzz and the phaser preserve the second input's eigenbasis and that neither of the operations is commutative.

Neither phaser nor fuzz are completely positive maps. Therefore they are not internal to the category of CPM(**FHilb**). In other words, they do not exist in our category of choice. Coecke and Meichanetzidis (2020) provide a richer category to which both operations are internal by applying a second CPM construction to CPM(**FHilb**) (explored in Ashoush, 2015). For this thesis, we will remain using

positive operators. We will thus ignore this observation, knowing that there are categorical constructions to overcome these objections.

3.5.5.4 Diag

Definition 20. *Let* A, B *be positive operators, then*

$$diag(\mathsf{A}, \mathsf{B}) \coloneqq \mathrm{dg}(\mathsf{A}) \circ \mathrm{dg}(\mathsf{B})$$

where $\mathrm{dg}(\mathsf{X})$ *sets all non-diagonal entries of the matrix* X *to 0.*

As *diag* sets all non-diagonal entries to 0, the outcome of *diag* will always be in the computational basis. This is an undesirable property, and we will see that *diag* does not perform well in our experimental evaluation (see Section 4.5).

3.5.6 Normalisation

To restrict the magnitudes of the entailment measures, the positive operators have to be normalised. Bankova et al. (2019) propose two normalisation conventions; (1) normalising the positive operators to trace 1 and (2) normalising the operators to a maximum eigenvalue of at most 1. As pointed out by Van de Wetering (2016), the first option trivialises k_{hyp} for $k = 1$ (i.e. crisp Löwner order), in that $\mathsf{A} \sqsubseteq_{k=1} \mathsf{B} \Rightarrow \mathsf{A} = \mathsf{B}$. Therefore we will follow Lewis (2020) in choosing normalisation to maximum eigenvalue of 1. This choice guarantees particularly nice properties for the Löwner order. In particular, the maximally mixed state is the bottom element and all pure states are maximal (Van de Wetering, 2018). This normalisation also guarantees that Lewis's (2020) logical negation operation (see Section 4.2.1.1) preserves positivity.

"Not a chicken"

4

Conversational negation of words

Contents

4.1 Intuition

As mentioned in the introduction, the conversational negation of words not only denies information, it elicits alternatives (Oaksford & Stenning, 1992). This search for alternatives builds on an intuitive understanding of the world that most humans possess. Let us come back to the example from the introduction:

a) `This is not a chicken; this is a goose.`

b) `This is not a chicken; this is a spaceship.`

We recall that the second sentence feels strange. It does not match the alternatives a listener considers when talking about `not chicken`.

The core hypothesis for our thesis is:

conversational negation is context dependent

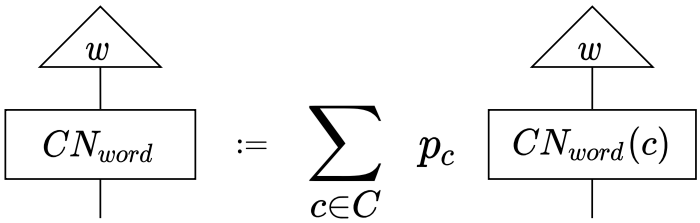
This reflects that `not chicken` in the context of `animals` elicits other alternatives than in the context of `meat`. Our framework has to capture the ambiguity of the different contexts which could have been assumed for the negation. These contexts depend on multiple factors, including the surrounding text as well as the understanding of the world that a speaker assumes a listener to have. Our framework will capture this intuitive understanding of the world in an ambiguous mixture of all contexts in which the negated word could occur. After the negation, we rely on the disambiguation of the DisCocirc framework to disambiguate the negation through the context contained in the text.

To model the negation, we additionally observe that negation can be viewed as an operation; if we know the meaning of some word w, we can derive the meaning of `not` w. Thus, if I define a new concept in this thesis, you can derive the meaning of its negation without me having to explain it explicitly.

4.2 The framework

The interpretation of the negation depends on the context. This context informs the possible alternatives. Thus our framework has to consider all possible contexts. For this, we utilise ambiguity. Ambiguity allows us to encode the possible interpretations of the negation in a single operation via a weighted mixture. Conversational negation of a word becomes a weighted mixture of the different interpretations, where each interpretation represents a negation under a different context. The weight of each element in the mixture then represents how likely an individual context is. Therefore

the framework for conversational negation of words is:



where C is the set of all contexts, $CN_{word}(c)$ is the negation of w under the context $c \in C$ and p_c is the weight of context c.

This leaves us with two tasks; determining the operation inside the negation box under a given context and determining the different contexts and their associated weights.

4.2.1 Filling the negation box

To model the conversational negation with its search for alternatives, we take the view of Prado and Noveck (2006). They propose that negation takes two stages; initially only denying information and secondly analysing the negation for alternatives. We thus need two ingredients; a logical negation to model information denial and a way to inform the search for alternatives.

4.2.1.1 Logical negation

Logical negation, which we will denote by $\neg$, occurs in many fields of mathematics. In a sense, the logical negation of some word `w` should capture everything that is not `w`. This section will explore four candidate operations for the logical negation; one based on the additive inverse, introduced by Lewis (2020) and three based on the multiplicative inverse, presented by us, in Rodatz et al. (2021). All operations are defined for positive operators, with which the later experimental validation will be conducted (see Section 4.5). Nevertheless, the general idea of the framework applies to other meaning spaces as long as they provide a logical negation and a method to encode ambiguity.

To assess our choices of logical negation theoretically, we will consider two important properties of logical negation in classical logic; the double negative and the contrapositive. The double negative states:

$$\neg(\neg P) = P$$

for some P, i.e. something that is "`not not a dog`" is a dog. In a conversation, the double negation often conveys a slightly different meaning than the simple, positive statement. However, as we are considering logical negation operations, this is an obvious property.

The contrapositive states:

$$P \sqsubseteq Q \iff \neg Q \sqsubseteq \neg P$$

for some P, Q. We have that every dog is an animal; therefore, something not being an animal means it cannot be a dog either. The contrapositive requires the logical negation to interact well with the entailment measures. As our entailment measures are graded, we will generalise the contrapositive to:

$$P \sqsubseteq_k Q \iff \neg Q \sqsubseteq_{k'} \neg P$$

where optimally $k = k'$.

We will analyse the four negation operations with respect to the desired properties of double negation and contrapositive.

Subtraction-from-identity-negation – Lewis (2020) introduces and experimentally validates a logical negation operation for positive operators.

Definition 21 (Lewis, 2020)**.** *The* ***subtraction-from-identity-negation****, called* $\neg_{sub}$*, is a unitary function from positive operators to positive operators. Let* X *be a positive operator, then we define*

$$\neg_{sub}\mathsf{X} \coloneqq \mathbb{I} - \mathsf{X}$$

where $\mathbb{I}$ *is the identity matrix with the same dimensions as* X*.*

This operation preserves positivity of operators (given the right choice of normalisation. See Section 3.5.6) and, in the case of projectors, maps to their orthonormal subspace. Therefore this operation is similar to proposals to logically negate vectors by mapping them to the orthogonal subspace (Widdows & Peters, 2003). This logical negation satisfies the double negative (see Theorem 3 in the Appendix). It also satisfies the graded contrapositive for k_{BA} (Lewis, 2020). It satisfies the contrapositive for k_{hyp} when $k_{\mathsf{hyp}} = 1$, i.e. crisp Löwner order (see Theorem 5 in the Appendix). Therefore it also satisfies the contrapositive for k_{E} when $k_{\mathsf{E}} = 1$ (see Theorem 6 in the Appendix). However, Lewis (2020) shows that it does not satisfy the contrapositive for k_{E} in general.

Inverse-negations – We introduced these multiplicative inverses in Rodatz et al. (2021) based on the observation that the matrix inverse reverses Löwner order (Baksalary et al., 1989). As the matrix inverse is not well defined for all positive operators, we propose two generalisations, respectively acting on the matrix's support and kernel and then combining them into one operation, in total, giving us three negation operations.

We recall from Section 3.5.2 that the kernel of a matrix is the subspace of vectors mapped to 0 under M. It is spanned by all eigenvectors with eigenvalue 0. The support is the set of vectors mapped to something other than 0 under M.

Definition 22 (Rodatz et al., 2021)**.** *The* ***support-inverse-negation****, called* $\neg_{supp}$*, is a unitary function from positive operators to positive operators. Let* X *be a positive operator with spectral decomposition* $\mathsf{X} = \sum_i \lambda_i |i\rangle \langle i|$*, then we define:*

$$\neg_{supp}\mathsf{X} := \sum_i \begin{cases} \frac{1}{\lambda_i} |i\rangle \langle i| & \text{if } \lambda_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

Thus the *support-inverse-negation* calculates the inverse of the support and leaves the kernel unchanged. It is equal to the Moore-Penrose generalised matrix inverse. It equals the normal matrix inverse when the kernel is empty. The *support-inverse-negation* satisfies the contrapositive for k_{hyp} when $rank(\mathsf{A}) = rank(\mathsf{B})$ (see Theorem 7 in the Appendix) and for k_{BA} when the two matrices are invertible and

have the same eigenbasis (see Corollary 1 in the Appendix). It does not satisfy the contrapositive for k_{E} (see Section A.1.2.2 in the Appendix).

The motivation behind the logical negation is that the logical negation of w captures everything that is not w. However, the *support-inverse-negation* leaves the kernel of the matrix untouched. Therefore it does not match that intuition, as some word w' whose support lives in the kernel of w, will still be in the kernel of $\neg_{supp} w$. Therefore we define the *kernel-inverse-negation* next to the *support-inverse-negation*.

Definition 23 (Rodatz et al., 2021)**.** *The* ***kernel-inverse-negation****, called* $\neg_{ker}$*, is a unitary function from positive operators to positive operators. Let* X *be a positive operator with spectral decomposition* $X = \sum_i \lambda_i |i\rangle \langle i|$*, then we define:*

$$\neg_{ker} X \coloneqq \sum_i \begin{cases} 0 & \text{if } \lambda_i > 0 \\ 1 & \text{otherwise} \end{cases}$$

The *kernel-inverse-negation* captures everything that is in the kernel of the operator being negated; it is defined as the identity over the kernel. It is equal to the limit of normalising the *support-inverse-negation* after setting all zero-valued eigenvalues to ϵ for $\epsilon \downarrow 0$. However, it does not preserve the information of the support. Thus any two matrices with the same kernel will have the same inverse, independent of their other eigenvalues. This also means that applying the *kernel-inverse-negation* twice leads to the maximally mixed state over the support. It therefore does not satisfy the double negative (see counterexample A.1.1.3). The *kernel-inverse-negation* satisfies the contrapositive for all entailment measures when $k = 1$ (see Corollary 2, Theorem 10 and Theorem 11 in the Appendix).

Intuitively the negation of a word should contain both elements close to the original words (i.e., elements in the support) and elements far away from the original word (i.e., elements in the kernel). Therefore the logical negation of a word should have non-zero values in the support and the kernel of the negated matrix. As neither the *support-inverse-negation* nor the *kernel-inverse-negation* fulfill that property, we propose one final logical negation, which combines both negations. We have:

Definition 24 (Rodatz et al., 2021)**.** *The* ***inverse-negation****, called* $\neg_{inv}$*, is a unitary function from positive operators to positive operators. Let* X *be a positive operator with spectral decomposition* $X = \sum_i \lambda_i |i\rangle \langle i|$*, then we define:*

$$\begin{aligned} \neg_{inv} X &:= normalise(\neg_{supp} X) + \neg_{ker} X \\ &= \sum_i \begin{cases} \frac{\lambda_{min}}{\lambda_i} & \text{if } \lambda_i > 0 \\ 1 & \text{otherwise} \end{cases} \end{aligned}$$

where λ_{min} *is the smallest non-zero eigenvalue.*

By normalising the *support-inverse-negation*, we guarantee that the outcome is still normalised correctly. Furthermore, we normalise before the addition to ensure that the smallest eigenvalue in the support will have the same value as the eigenvectors in the kernel, therefore weighing both negations equally. While the *inverse-negation* guarantees values in both the kernel and support of the negated matrix, it has some curious properties. For example, a matrix with eigenvalues only 1 and 0 will become the maximally mixed state. It also does not satisfy the double negative (see counterexample A.1.1.4) or any of the contrapositives (see Section A.1.2.4 in the Appendix).

Comparing the negations – We have introduced four different forms of logical negation, one based on the additive inverse and three based on the multiplicative inverse. Each fulfill some of the desired properties. However, none of them fulfill all. The following table gives an overview:

Table 4.1: Theoretical analysis of the properties of the proposed logical negation operations

		$\neg_{sub}$	$\neg_{supp}$	$\neg_{ker}$	$\neg_{inv}$
Double negation		✓	✓	✗	✗
Contra-positive	k_{hyp}	✓ for $k = 1$	✓ when $rank(\mathsf{A}) = rank(\mathsf{B})$	✓ for $k = 1$	✗
	k_E	✓ for $k = 1$	✗	✓ for $k = 1$	✗
	k_{BA}	✓	✓ for same e.b. and invertible	✓ for $k = 1$	✗

The orange ticks hold only under certain conditions. We note that while the *inverse-negation* has no nice theoretical properties, it at least captures elements in both the kernel and support of the negated operator. We will see that the *subtraction-from-identity-negation* not only has decent theoretical properties but also outperforms the other negations in our experiments (Section 4.5). Chapter A.1 in Appendix A contains all the proofs and counter examples for each of the properties in Table 4.1 as already mentioned throughout this section.

We observe that logical negation is not linear (intuitively $\neg(A \wedge B) \neq \neg A \wedge \neg B$). Thus no linear map and therefore no morphism in CPM(**FHilb**) can capture the logical negation[1]. Indeed none of our proposed operations are linear. Therefore none of them exist in our meaning category CPM(**FHilb**). We will have to step out of our category to perform our calculations. It is left for future work to devise other categories which allow for native logical negation operations.

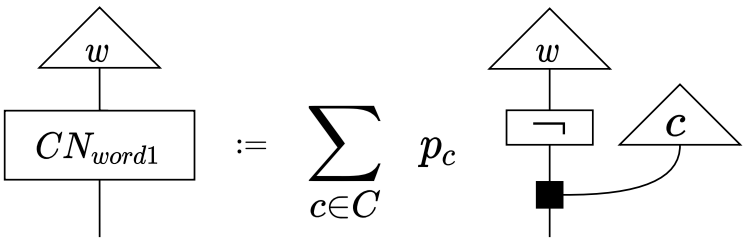
4.2.1.2 A first framework

Having defined possible operations to model logical negation, we propose to model conversational negation of words w under a given context c in two steps:

1. Take the logical negation of w to model denial of information
2. Update the result with the context c to inform the search for alternatives

This aligns with Prado and Noveck's (2006) view on how humans perceive negation. As mentioned earlier, to deal with the different contexts, we will take a weighted sum over all possible contexts.

Diagrammatically the framework, called CN_{word1}, looks as follows:



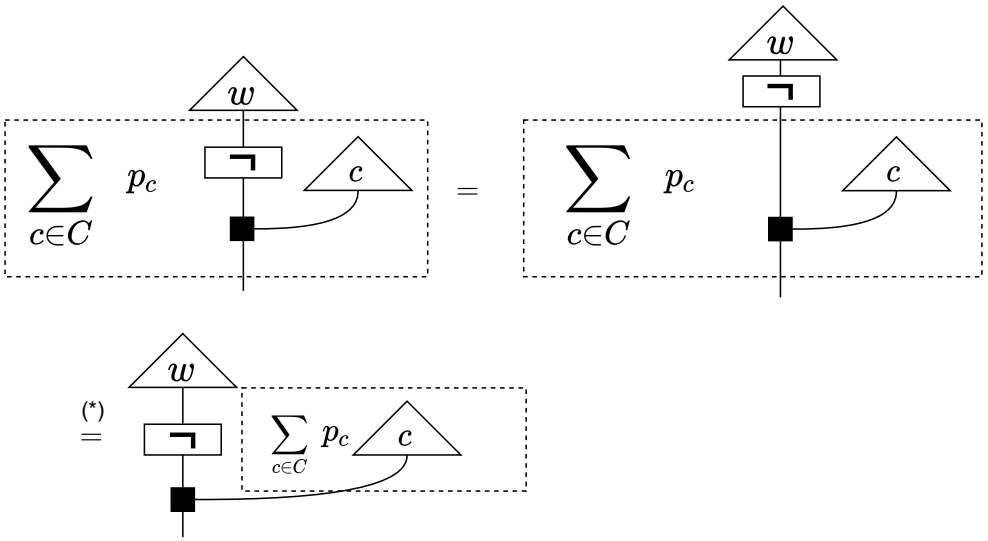
[1]We recall that morphisms in CPM(**FHilb**) were linear maps, which preserve positivity.

In a sense, each summand in the framework can be understood as saying "`It is not a` w `but it is still a` c" (i.e. "`It is not a dog but it is still an animal.`"). We then sum over all these sentences.

The framework is flexible to the choice of logical negation and update operation. In Section 4.5, we will experimentally compare different choices and their interactions.

4.2.2 Pre-computing a worldly context

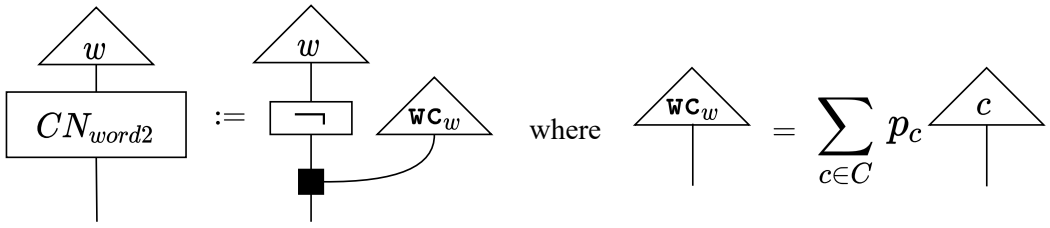
We can utilise the diagrammatic calculus to rewrite our framework as:



(4.1)

where the dotted box signals the scope of the sum. The first equality holds as the negation of w is not directly affected by the sum. The second equality, marked by a $*$, does not always hold. This is since some of the composition operations are basis dependent (the *spider, fuzz* and *phaser*). The equality does not hold when the basis of the composition is dependent on the context[2].

We thus define a second framework, which we call CN_{word2}:



[2]Both, the star and the dotted box to mark the scope of the sum, are only necessary because some of the operations depend on the context. If we fix the operations (and do not adapt their basis depending on the context), these derivations would be trivial.

This is the framework for conversational negation proposed by us in Rodatz et al. (2021). This second framework is equivalent to the first one for some but not all choices of composition. Additionally to different results, it provides a different view on the negation. Instead of summing over all negations under different contexts, we have one *worldly context*, $\texttt{wc}_w$, which captures a weighted sum of all contexts in which the word could appear. Thus instead of summing over different interpretations, this framework corresponds to saying "`It is not` w `but it must still occur in the context, in which we know` w `to usually occur.`" Where the context, in which w usually occurs is captured by the worldly context of w called $\texttt{wc}_w$.

The advantage of this framework is that the context becomes a single meaning state, which we can pre-compute. This simplifies the language circuits by hiding the sum and reduces computation. We will explore both frameworks experimentally in Section 4.5. However, before we can apply either framework, we must determine the contexts in which a word appears and their respective weights.

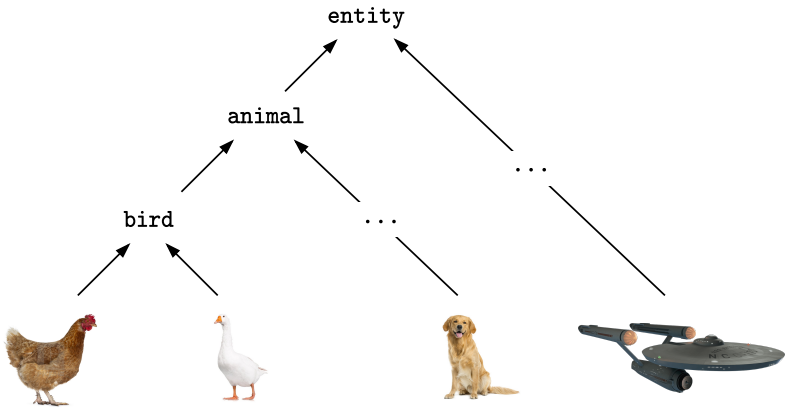
4.3 Determining the context

In addition to filling the negation box, we must determine the contexts in which a word occurs and their respective weights. These contexts encode the listeners understanding of the world, which is not explicitly present in a given text. The motivation behind these contexts is to re-introduce this knowledge to the framework. The contexts should capture and grade all situations in which a particular word can occur, which are then combined via a weighted mixture, i.e. ambiguity. Depending on the present text, the appropriate choice of context should then occur after the negation during further meaning updates utilising the process of disambiguation as presented in Coecke and Meichanetzidis (2020). Our experimental validation in Section 4.5 focuses on validating correlations between alternatives to a negation elicited by a human reader and alternatives elicited by our frameworks. Experiments on the later disambiguation of the negation require experiments that go beyond individual sentences and are left for future work.

We will propose two methods for deriving context; (1) utilising external entailment hierarchies and (2) from the entailment information encoded in the positive operators. Our experimental validation will mainly focus on the first approach but provide some insights into the potential feasibility of the second.

4.3.1 Context from entailment hierarchies

One method to derive context from an external source is utilising entailment hierarchies such as the human-curated WordNet (Miller, 1995) or the unsupervised Hearst patterns (Hearst, 1992). Entailment hierarchies provide for each word w several more general words of which w is a type. These more general words grow in scope, each encompassing the previous until eventually reaching the broadest word. In the case of WordNet, this is `entity`. For the word `chicken`, WordNet would provide an entailment structure such as:



where each arrow represents an entailment relation. All entailment relations taken together form a tree, with `entity` being the root. We thus obtain a directed path from the word `chicken` to `entity`. From this path we learn that each `chicken` is a `bird`, each `bird` is an `animal`, and each `animal` is an `entity`. We use this information to derive the context in which a word is most likely to appear. We usually think of `chicken` in the context of `bird`. Sometimes we think of it in the context of `animal`, and even less frequently, we think of it in the context of `entity`. Thus something that is not a `chicken` is most likely to be another `bird`, such as a `goose`. It is slightly less likely to be another `animal` such as a `dog`. It is even less

likely to be another `entity` such as a `spaceship`. We propose to build the context of a word out of the elements in this entailment hierarchy, where words lower in the hierarchy have a higher weight than more distant words.

Let $h_1, ..., h_n$ be such a path from some word w to the word `entity` in the entailment hierarchy ordered from closest to furthest. Then we define the context of w to be:

$$C = \{h_1, ..., h_n\}$$

and restrict the weights such that:

$$\forall i, j \in \{1, ..., n\}, \quad i < j \quad \iff \quad p_{h_i} > p_{h_j}$$

In Section 4.5 we will experimentally explore various gradings on the weights.

We assume that a positive operator for a word w encodes all the meanings of the words that are a type of w. Thus the positive operator for `animal` encodes `bird, chicken, goose` and `dog`. In the experiments, we guarantee this through the way we construct the positive operators (see Section 4.5.2).

One observation is that `chicken` can refer to different meanings, such as `chicken` the food and `chicken` the animal. WordNet captures these meanings in *SynSets* each having their own entailment hierarchy towards `entity`. For example, we could get a second hierarchy `chicken` → `meat` → `food` → `entity`. We propose taking the union over all the hierarchies, providing one big context for all possible meanings. The disambiguation through meaning updates of the surrounding text determines the correct interpretation via later updates, similar to the disambiguation illustrated in Coecke and Meichanetzidis (2020). While the mechanics of meaning updates suggest this works, these assumptions have to be experimentally validated. This is left for future work.

4.3.2 Context from positive operator entailment

One challenge with WordNet is that it does not contain graded entailment. Properties such as "`most dogs are pets`" are therefore not quantified within WordNet.

While there are proposals to extend WordNet to include such information (Ahsaee et al., 2014; Boyd-Graber et al., 2005), they have not been implemented yet.

One property of positive operators is that they can encode graded entailment information. Thus instead of relying on external sources, we can extract the entailment information encoded in the positive operators.

To utilise the entailment information encoded in the positive operators, we propose to generalise the idea mentioned in the previous section. Instead of having an entailment tree as provided by WordNet, we can build a directed, weighted graph. In this graph each word corresponds to a vertex and each edge (w_1, w_2) corresponds to an entailment relation. The weight of the edge (w_1, w_2) is the value of the graded entailment $w_1 \sqsubseteq_k w_2$. For any word pair (w_1, w_2) we thus get two values; p and q such that $w_1 \sqsubseteq_p w_2$ and $w_2 \sqsubseteq_q w_1$. They correspond to the weights on the respective edges from w_1 to w_2 and from w_2 to w_1. In an idealised setting, words are in one of three relations:

- They are synonyms - in this case, both p and q are high
- They are not related - in this case, both p and q are low
- One contains the other - in this case, one of p and q is high, while the other is low (every dog is an animal but not every animal is a dog)

We can thus see that the relation of w_1 and w_2 is dependent on both p and q.

To calculate the contexts of some word w, we have to consider all other words $w_1, ...w_n$ to which w is connected. We will thus have the weights $p_1, ..., p_n$ and $q_1, ..., q_n$ on the connecting edges, respectively from w to w_i and from w_i to w. We then define:

$$C = \{w_1, ..., w_n\}$$

and the respective weights are:

$$p_{w_i} = f(p_i, q_i)$$

where f is some function of the weights p_i and q_i.

4.3.3 A toy example

We will give a toy example to see how the frameworks work in action. For this toy example, we will use the *subtraction-from-identity-negation*, $\neg_{sub}$, as logical negation and the *spider* as composition. For simplicity, we will assume that our words are pure states, such that all operations can be done in the computational basis. This means that the basis of the composition does not change depending on the context. Therefore the equalities in Equation 4.1 hold, and both proposed frameworks are identical. For this example, we will use CN_{word2} as it requires fewer computations. For CN_{word2}, we first calculate the weighted sum of all contexts to get the *worldly context* capturing all contexts in which the word usually occurs. The conversational negation is then calculated with respect to this worldly context.

Let us say we want to find the negation of the words `chicken` to derive the meaning of the sentences "`This is not a chicken`". Let us additionally assume that $\{[\![\texttt{chicken}]\!], [\![\texttt{goose}]\!], [\![\texttt{dog}]\!], [\![\texttt{spaceship}]\!]\}$ are pure states, which form the orthonormal basis of the meaning space we are working in. In practice, the orthonormal basis is usually much larger, and our meanings are not orthogonal. These simplifications are for the sake of this toy example to give an intuition about the framework.

We first have to calculate the worldly context of the words `chicken`. Similar to the earlier example in Section 4.3.1 we will assume that the entailment tree of `chicken` is made up out of `bird` $\rightarrow$ `animal` $\rightarrow$ `entity` with:

$$\begin{aligned}
[\![\texttt{bird}]\!] =& \frac{1}{2}[\![\texttt{chicken}]\!] + \frac{1}{2}[\![\texttt{goose}]\!] \\
[\![\texttt{animal}]\!] =& \frac{1}{3}[\![\texttt{chicken}]\!] + \frac{1}{3}[\![\texttt{goose}]\!] + \frac{1}{3}[\![\texttt{dog}]\!] \\
[\![\texttt{entity}]\!] =& \frac{1}{4}[\![\texttt{chicken}]\!] + \frac{1}{4}[\![\texttt{goose}]\!] + \frac{1}{4}[\![\texttt{dog}]\!] \\
&+ \frac{1}{4}[\![\texttt{spaceship}]\!]
\end{aligned}$$

The worldly context of chicken is then made up of a weighted sum of these contexts, where the closer contexts are weighted higher than the contexts further

away. For example, we would have:

$$\begin{aligned}[\![\texttt{wc}_{\texttt{chicken}}]\!] =& \frac{1}{2}[\![\texttt{bird}]\!] + \frac{1}{3}[\![\texttt{animal}]\!] + \frac{1}{6}[\![\texttt{entity}]\!] \\ \approx& \frac{2}{5}[\![\texttt{chicken}]\!] + \frac{2}{5}[\![\texttt{goose}]\!] + \frac{3}{20}[\![\texttt{dog}]\!] \\ &+ \frac{1}{20}[\![\texttt{spaceship}]\!]\end{aligned}$$

We can see that the worldly context of `chicken` contains `chicken`. This is logical, as the contexts in which `chicken` usually occurs by definition contain `chicken`.

But then, to calculate the conversational negation of `chicken`, we first have to apply $\neg_{sub}$ to $[\![\texttt{chicken}]\!]$. We get:

$$\neg_{sub}([\![\texttt{chicken}]\!]) = [\![\texttt{goose}]\!] + [\![\texttt{dog}]\!] + [\![\texttt{spaceship}]\!]$$

In the second step, we combine the logical negation with the worldly context. Using the *spider* in the computational basis, we get:

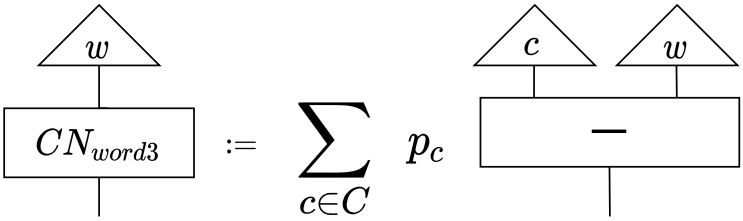
$$\mathit{spider}(\neg_{sub}([\![\texttt{chicken}]\!]), [\![\texttt{wc}_{\texttt{chicken}}]\!]) = \frac{2}{5}[\![\texttt{goose}]\!] + \frac{3}{20}[\![\texttt{dog}]\!] + \frac{1}{20}[\![\texttt{spaceship}]\!]$$

We see that the final result not only contains all entities which are not `chicken`, but it also preserves the proportions of the worldly context.

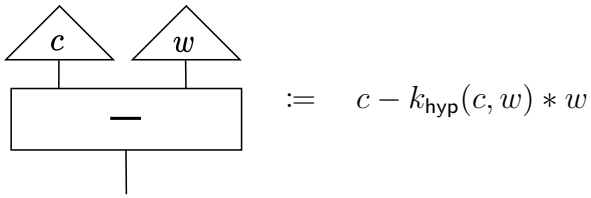
4.4 More negation frameworks

The motivation for the framework, which we originally introduced in Rodatz et al. (2021) was based on the psychological observations by Prado and Noveck (2006). We will additionally present a second framework. Instead of first negating and then updating with the context, we will negate with respect to the context. This is akin to Hermann et al.'s (2013) proposal of giving each vector a domain and a value within that domain, where the negation leaves the domain untouched and only affects the value. The main difference is that we will sum over various contexts; therefore, in a sense, consider multiple domains.

For do this, we will generalise the *subtraction-from-identity-negation* to a subtraction from the context. We define:

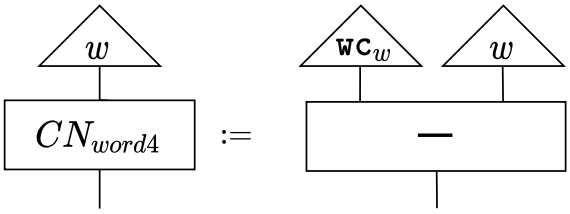


where the minus-operation is defined as:



We thus define the negation under a given context as subtracting the word from the context. We scale w by k_{hyp}, as $k_{\mathsf{hyp}}(c, w)$ is the maximal value such that $c - k_{\mathsf{hyp}}(c, w) * w$ is positive. Thus by utilising k_{hyp}, we ensure that the outcome of our negation is indeed positive.

We can similarly rewrite this framework to get a second proposal based on our intuition with the previous framework. Once more, we rewrite the framework first to compute the worldly context and then apply the negation. Therefore, we do not sum over the outcomes of the negation. Instead, we first sum over the contexts and then compute the negation. This change will yield a different result, as the k_{hyp} calculation is done after the summation of the contexts. We thus get a fourth framework:



4.4.1 A toy example - reprise

We will once more negate `chicken` to illustrate the new frameworks. In particular, we will illustrate the workings of CN_{word4}.

We recall that the pure states $[\![\texttt{chicken}]\!], [\![\texttt{goose}]\!], [\![\texttt{dog}]\!]$ and $[\![\texttt{spaceship}]\!]$ form the orthonormal basis of the meaning space we are working in. Our calculations for the worldly context of `chicken` gave us:

$$[\![\texttt{wc}_{\texttt{chicken}}]\!] \approx \frac{2}{5}[\![\texttt{chicken}]\!] + \frac{2}{5}[\![\texttt{goose}]\!] + \frac{3}{20}[\![\texttt{dog}]\!] + \frac{1}{20}[\![\texttt{spaceship}]\!]$$

But then to calculate the negation of `chicken` under CN_{word4} we first have to calculate $k_{\texttt{hyp}}([\![\texttt{wc}_{\texttt{chicken}}]\!], [\![\texttt{chicken}]\!])$. We have:

$$[\![\texttt{wc}_{\texttt{chicken}}]\!] - \frac{2}{5}[\![\texttt{chicken}]\!] = \\ (\frac{2}{5}[\![\texttt{chicken}]\!] + \frac{2}{5}[\![\texttt{goose}]\!] + \frac{3}{20}[\![\texttt{dog}]\!] + \frac{1}{20}[\![\texttt{spaceship}]\!]) - \frac{2}{5}[\![\texttt{chicken}]\!] \geq 0$$

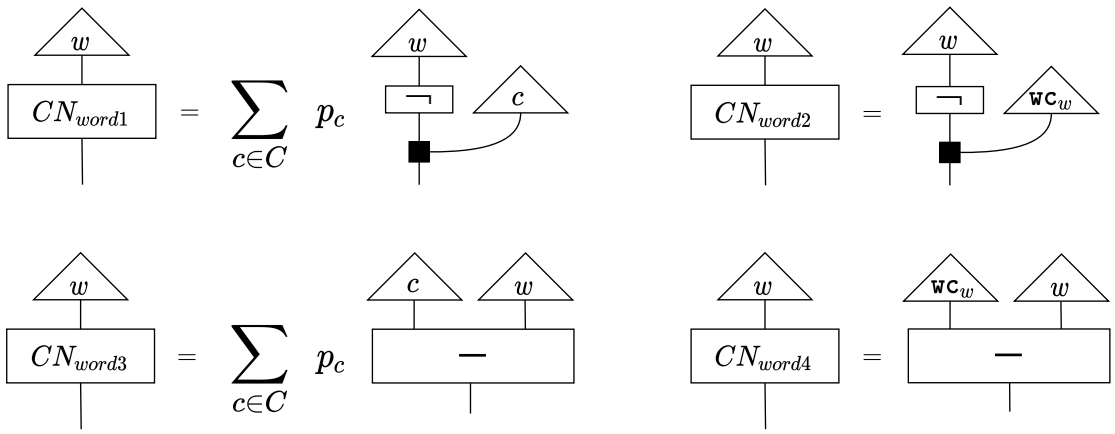
For any value higher than $\frac{2}{5}$ the outcome would not be positive anymore, as we would have a negative value for $[\![\texttt{chicken}]\!]$. Thus we can conclude that $k_{\texttt{hyp}}([\![\texttt{wc}_{\texttt{chicken}}]\!], [\![\texttt{chicken}]\!]) = \frac{2}{5}$. But then we can compute the negation of `chicken` as:

$$\begin{aligned} CN_{word4}([\![\texttt{chicken}]\!]) &= [\![\texttt{wc}_{\texttt{chicken}}]\!] - \frac{2}{5}[\![\texttt{chicken}]\!] \\ &= \frac{2}{5}[\![\texttt{goose}]\!] + \frac{3}{20}[\![\texttt{dog}]\!] + \frac{1}{20}[\![\texttt{spaceship}]\!] \end{aligned}$$

In this simplified scenario, the negation of `chicken` under CN_{word4} is identical to the negation under CN_{word2}.

4.5 Experimental validation

We have defined four different frameworks for the conversational negation of words. In this section, we will run experiments to validate and compare our proposals. The four frameworks are:



For our experiments, we use the dataset provided by Kruszewski et al. (2016)[3]. Kruszewski et al. created over a thousand word pairs which create sentences with a negation. They then asked human participants to rate these sentences on how plausible they are to occur in a normal conversation. They proposed various methods to predict the plausibility rating, some of which achieve a high correlation with human intuition. However, they do not present a framework to calculate a meaning representation of the result of conversational negation.

We claim that our proposed frameworks calculate a meaning representation of conversational negation. To check whether the results of our frameworks are sensible, we will check if they match human intuition. We will therefore check if the alternatives elicited by our frameworks correlate to the alternatives humans consider. In contrast to Kruszewski et al., our primary goal is not to predict plausibility. Instead, we use the predicted plausibility ratings to verify our framework. We observe that we expect an operation for conversational negation to perform well in these experiments. However, a good performance is not sufficient to prove that an operation does model conversational negation. Doing well at these experiments is a necessary but not sufficient condition. We therefore do additional data exploration in Section 4.6.

[3]The data set is available at http://marcobaroni.org/PublicData/alternatives_dataset.zip.

4.5.1 Dataset

Kruszewski et al.'s (2016) dataset contains 1231 pairs of nouns (w_N, w_A) of a word to be negated w_N and an alternative w_A. These word pairs were made into sentences of the form "`This is not a` w_N`, it is a` w_A.". For example "`This is not a lemon, it is a truth.`".

The word pairs were created by picking 50 common nouns as w_N. Kruszewski et al. then consulted various sources to find possible alternatives for w_N, ranging from synonyms to random words. Finally, they gave these sentences to human participants to rate them on a scale from 1 to 5 on how plausible they are to appear in a human conversation (with 1 being very implausible and 5 being very plausible).

4.5.2 Methodology

4.5.2.1 Building positive operators

We build positive operators from GloVe vectors of dimension 50. GloVe vectors are vector representations for words. They are built by an unsupervised algorithm from large text corpora via co-occurrence (Pennington et al., 2014). The vectors we use were trained on Wikipedia and newspaper articles.

To build the positive operators, we utilise the method proposed by Lewis (2019). Following Lewis's example, we use the entailment data found in WordNet. To calculate the positive operator for some word w, we take the following steps:

1. Find all words that are a type of w, let us call them $w_1, ..., w_n$
2. Find their corresponding GloVe vectors $v_w, v_{w_1}, ..., v_{w_n}$
3. Calculate $[\![w]\!] = \sum_{v \in \{v_w, v_1, ..., v_n\}} |v\rangle \langle v|$

where we use the fact that we can obtain positive operators from vectors by taking the outer product. Thus, the positive operator of w contains the vector for w and the vectors for words that are a type of w. For example, we build the positive operator for `animal` from the vectors for `animal, dog, golden retriever, bird, ...`. Following Lewis, we weigh all vectors equally. We could probably obtain

better results in our experiments with more complicated methods for building positive operators. For example, such methods could consider the distance to the word w in the entailment hierarchy. However, as the goal of the experiments is mainly to validate and compare our conversational negation frameworks, we did not explore this further.

4.5.2.2 The experiments

Having built the positive operators, for each word pair (w_N, w_A), we calculate the conversational negation of w_N and then calculate the similarity with w_A. We consider these similarity measures as the plausibility for w_A to be a valid alternative to `not` w_N. We then compare these scores with the human plausibility ratings via the Pearson correlation. The code with which the experiments were conducted is available at *link to personal GitHub page redacted - the code is available with the submission*. The results will be analysed with respect to the framework and choice of operation for the frameworks. Additional analysis will concern the choice of context grading and similarity measure.

Framework and operation comparison

In our experiments, we compare the four frameworks proposed in this chapter. Additionally, as a baseline, we display the correlations obtained from comparing the w_N with w_A (labelled w_N) and the correlation of comparing the worldly context of w_N with w_A (labelled $\texttt{wc}_{w_N}$).

For the first two negation frameworks, we compare the four logical negations $\neg_{sub}$, $\neg_{supp}$, $\neg_{ker}$ and $\neg_{inv}$, and the four composition operations *spider, fuzz, phaser* and *diag*. For the basis dependent operations (*spider, fuzz* and *phaser*) we try both the basis of w_N - the word being negated (referred to as 'w') and the basis of the context (referred to as 'c').

Context comparison

For all frameworks, we utilise WordNet to calculate the context, as proposed in Section 4.3.1. As outlined in Section 4.3, we want to weigh context words that are further away from the negated word less than closer words. We explore different monotone decreasing weight functions based on the distance to the word. For a word w with the hypernyms $h_1, ..., h_n$ ordered from closest to furthest, we compare:

$$p_{h_i} = \texttt{poly}_x(i) \coloneqq (n-i)^x$$
$$p_{h_i} = \texttt{exp}_x(i) \coloneqq (1 + \frac{x}{10})^{(n-i)}$$
$$p_{h_i} = \texttt{hyp-kE}_x(i) \coloneqq (n-i)^{\frac{x}{2}} k_{\mathsf{E}}(w, h_i)$$
$$p_{h_i} = \texttt{hyp-khyp}_x(i) \coloneqq (n-i)^{\frac{x}{2}} k_{\mathsf{hyp}}(h_i, w)$$

The first two functions are simple polynomial and exponential functions. The second two functions are polynomial. However, the result is then multiplied by the entailment of the word and the respective context. This is similar to the proposal of Section 4.3.2.

In Rodatz et al. (2021), we only explored the first three context functions, but further experiments have shown that the last one performs particularly well. We have additionally explored other context functions combining other similarity measures proposed thought this thesis with monotone functions. However, we only present the most successful results.

For all of our context functions, a fixed x value has to be picked. The x parameter indicates to what degree distance in the entailment tree should be taken into account. We explore multiple. At $x = 0$ all contexts are weighted equally. For example $\texttt{poly}_0 = (n-i)^0 = 1$. Thus all contexts have the same weight, independent of their distance i. For higher x values, more distant contexts have comparably lower weights. For $\texttt{poly}_{10} = (n-i)^{10}$ smaller i values get a substantially larger weight than higher i values.

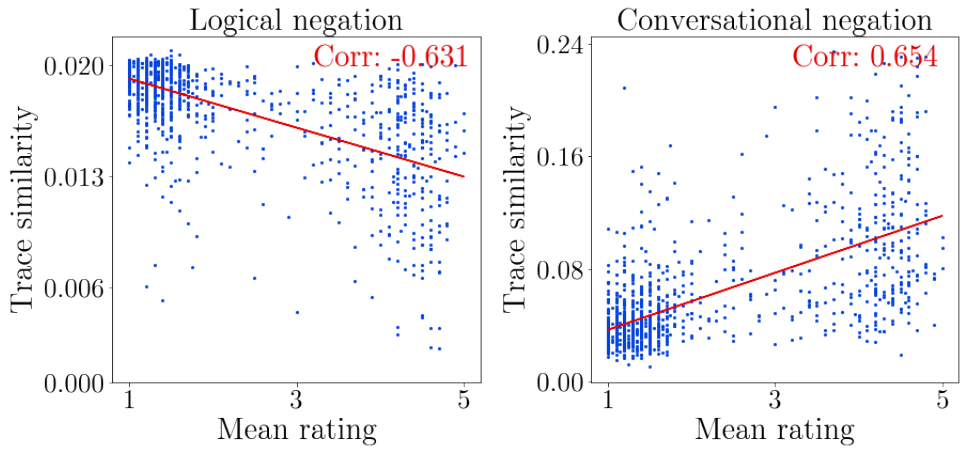


Figure 4.1: Correlation with human intuition of logical negation (left) and conversational negation CN_{word1} with *phaser* ('c'), $\neg_{sub}$ and sim_{trace} (right)

Similarity measure comparison

To find the plausibility rating, we calculate the similarity between the conversational negation of w_N and w_A. We use the three entailment measures $k_{\mathsf{hyp}}, k_{\mathsf{E}}$ and k_{BA} as well as the *trace similarity*. *Trace similarity* is defined as such:

Definition 25. *Let A, B be positive operators. Then the **trace similarity** of A and B is defined as:*

$$sim_{trace}(A, B) := \frac{dot(A, B)}{trace(A) \cdot trace(B)}$$

This is analogous to the cosine similarity for vectors, commonly used for similarity measures on vectors. We use this measure as a baseline on top of our entailment measures.

For the asymmetric measures, k_{hyp} and k_{E}, we calculate entailment in both directions. We will call the entailment from $CN_{word}(w_N)$ to w_A, k_{E1} and k_{hyp1} respectively and the entailment from w_A to $CN_{word}(w_N)$, k_{E2} and k_{hyp2} respectively. We will point out once more that we are not using the basic k_{hyp} but rather the generalisation proposed in Section 3.5.3, which does not enforce $supp(\mathsf{A}) \subseteq supp(\mathsf{B})$ for a non-zero entailment $\mathsf{A} \sqsubseteq_{k_{\mathsf{hyp}}} \mathsf{B}$.

4.5.3 Results

The analysis reveals that our best framework (CN_{word1} with $\neg_{sub}$, the *phaser* in the basis of w_N and sim_{trace}) achieves a statistically significant Pearson correlation of 0.654 with the human ratings when paired with the context function $\texttt{hyp-khyp}_4$. Figure 4.1 (right) shows for each word pair the similarity result on the x-axis vs the human rating on the y-axis.

In contrast to this, comparing the logical negation ($\neg_{sub}$) with the alternative via the trace similarity gives a negative correlation with human intuition (shown Figure 4.1 on the left). This negative correlation comes from logical negation capturing the opposite of the word. In a sense, it gives a result that is maximally far away from the original word. This contradicts Kruszewski et al.'s (2016) observation that the alternatives to a negation mostly appear in similar contexts as the original word. These results illustrate that simple logical negation does not capture the human intuition of conversational negation. However, upon amending the results with the worldly context, the correlation becomes positive.

We tested the four different frameworks with different choices of operations for the first two frameworks (as explained in the methodology). All the correlation results can be found in Table 4.2. The first two frameworks give the same result for many composition operations; they are displayed only once in those cases (those entries are labelled as framework $CN_{word1}\&CN_{word2}$). This table displays the correlation under the context function $\texttt{hyp-khyp}_4$ for which we get maximal values with CN_{word1} and CN_{word2}. Appendix B.1 shows the same correlations under the context function $\texttt{poly}_4$, for which CN_{word3} and CN_{word4} perform optimally. This is also the context function we chose in Shaikh et al. (2021). Therefore the results presented here differ from the ones presented in the paper. All correlations above 0.4 are highlighted in green. We will now explore this data for the different variables.

4.5.3.1 Framework comparison

All four frameworks achieve Pearson correlations above 0.55 (and above 0.58 for the other context function). However, CN_{word1} and CN_{word2} in the combination with

Table 4.2: Pearson correlation of different framework under context function $\texttt{hyp-khyp}_4$ with human intuition. Correlations above 0.4 are highlighted in green.

Framework	Logical negation	Composition	k_{E1}	k_{E2}	k_{hyp1}	k_{hyp2}	k_{BA}	sim_{trace}
w_N	—	—	0.464	0.551	0.303	-0.003	0.268	0.575
$\texttt{wc}_{w_N}$	—	—	0.409	0.569	0.300	0.407	0.303	0.651
CN_{word1} & CN_{word2} *(negations give same results under these composition operations)*	$\neg_{sub}$	$spider_{\mathrm{w}}$	-0.193	-0.259	0.285	0.253	0.247	-0.050
		$phaser_{\mathrm{w}}$	0.375	0.590	0.305	0.403	0.298	0.654
		$fuzz_{\mathrm{w}}$	-0.233	-0.111	0.299	0.212	0.261	0.468
		diag	-0.263	-0.281	0.300	-0.024	0.258	-0.077
	$\neg_{supp}$	$spider_{\mathrm{w}}$	0.167	0.295	0.232	-0.120	0.159	0.404
		$phaser_{\mathrm{w}}$	-0.151	0.149	0.246	0.080	0.148	0.185
		$fuzz_{\mathrm{w}}$	-0.186	0.062	0.244	0.059	0.146	0.019
		diag	-0.262	-0.009	0.217	0.060	0.131	-0.059
	$\neg_{ker}$	$spider_{\mathrm{w}}$	-0.251	-0.250	0.102	0.141	0.146	-0.463
		$phaser_{\mathrm{w}}$	0.284	0.413	0.310	0.293	0.201	0.579
		$fuzz_{\mathrm{w}}$	-0.228	-0.117	0.293	0.093	0.185	0.275
		diag	-0.248	-0.215	0.290	-0.049	0.182	0.001
	$\neg_{inv}$	$spider_{\mathrm{w}}$	-0.170	-0.039	0.237	0.047	0.124	0.157
		$phaser_{\mathrm{w}}$	0.305	0.453	0.312	0.186	0.193	0.587
		$fuzz_{\mathrm{w}}$	-0.223	-0.110	0.301	0.012	0.172	0.227
		diag	-0.256	-0.219	0.295	-0.044	0.173	-0.024
CN_{word1}	$\neg_{sub}$	$spider_{\mathrm{c}}$	-0.106	0.156	0.298	0.344	0.242	0.474
		$phaser_{\mathrm{c}}$	-0.265	-0.343	0.301	-0.260	0.265	-0.317
		$fuzz_{\mathrm{c}}$	-0.258	-0.245	0.301	-0.061	0.266	-0.071
	$\neg_{supp}$	$spider_{\mathrm{c}}$	0.226	0.330	0.240	0.218	0.159	0.448
		$phaser_{\mathrm{c}}$	-0.077	0.013	0.235	0.082	0.150	0.274
		$fuzz_{\mathrm{c}}$	-0.210	-0.024	0.231	0.025	0.153	0.068
	$\neg_{ker}$	$spider_{\mathrm{c}}$	-0.120	0.088	0.237	0.254	0.145	0.432
		$phaser_{\mathrm{c}}$	-0.267	-0.280	0.296	-0.208	0.183	-0.333
		$fuzz_{\mathrm{c}}$	-0.244	-0.202	0.295	-0.061	0.184	-0.083
	$\neg_{inv}$	$spider_{\mathrm{c}}$	-0.003	0.220	0.242	0.320	0.130	0.463
		$phaser_{\mathrm{c}}$	-0.262	-0.219	0.300	-0.101	0.177	-0.151
		$fuzz_{\mathrm{c}}$	-0.236	-0.194	0.300	-0.052	0.176	-0.065
CN_{word2}	$\neg_{sub}$	$spider_{\mathrm{c}}$	-0.155	0.025	0.301	0.197	0.211	0.284
		$phaser_{\mathrm{c}}$	-0.278	-0.302	0.309	0.095	0.233	-0.272
		$fuzz_{\mathrm{c}}$	-0.257	-0.168	0.309	0.101	0.236	-0.038
	$\neg_{supp}$	$spider_{\mathrm{c}}$	0.212	0.410	0.245	0.110	0.166	0.410
		$phaser_{\mathrm{c}}$	-0.043	0.069	0.248	-0.264	0.161	0.274
		$fuzz_{\mathrm{c}}$	-0.200	-0.010	0.231	0.030	0.153	0.076
	$\neg_{ker}$	$spider_{\mathrm{c}}$	-0.181	0.004	0.240	0.120	0.118	0.089
		$phaser_{\mathrm{c}}$	-0.271	-0.257	0.308	0.073	0.138	-0.325
		$fuzz_{\mathrm{c}}$	-0.203	-0.141	0.306	0.098	0.139	-0.035
	$\neg_{inv}$	$spider_{\mathrm{c}}$	-0.055	0.170	0.236	0.264	0.106	0.338
		$phaser_{\mathrm{c}}$	-0.270	-0.171	0.304	0.056	0.134	-0.178
		$fuzz_{\mathrm{c}}$	-0.186	-0.129	0.307	0.103	0.134	-0.034
CN_{word3}	—	—	0.113	0.297	0.301	0.247	0.169	0.574
CN_{word4}	—	—	0.117	0.249	0.302	0.135	0.140	0.556

the $phaser_w$ and $\neg_{sub}$ achieves the highest correlation of 0.654. Neither CN_{word3} nor CN_{word4} achieve a correlation of 0.6 or higher.

Both CN_{word2} and CN_{word4} are much faster computationally, given that compositions and entailment calculations only have to be done once per negation.

The analysis reveals that both the comparison between the word and its alternative (row w_N) and the comparison of the worldly context of w_N with the alternative (row $\texttt{wc}_{w_N}$) achieve very high correlations with the human intuition. As in distributional semantics, the negation of a word and the word itself appear in similar contexts (Mohammad et al., 2013; Oaksford & Stenning, 1992). For example `fast` and `slow` will both co-occur in contexts of `speed`, `racing` and `driving`. Therefore, they have very similar representations. Thus simply comparing the word with its alternative is a good indicator of the plausibility of the negation. Neither leaving the word as is nor the worldly context are plausible operations for modelling conversational negation. This once more reflects the point that while these experiments are necessary for any negation operation to pass, they are not sufficient to prove that it is indeed a valid operation for negation.

4.5.3.2 Operation comparison

The first two frameworks are dependent on the choice of operations. Therefore, we will compare the correlations of CN_{word1} and CN_{word2} for the choice of logical negation, composition and basis as they are displayed in Table 4.2.

Logical negation

We compare the *subtraction-from-identity-negation* ($\neg_{sub}$), *support-inverse-negation* ($\neg_{supp}$), *kernel-inverse-negation* ($\neg_{ker}$) and the *matrix-inverse-negation* ($\neg_{inv}$). $\neg_{sub}$ clearly outperforms the other negations in most cases. It is the only logical negation which achieves correlations above 0.6. It does this, despite not having optimal theoretical interaction with some of the entailment measures (see Table 4.1).

We recall that the logical negation $\neg_{inv}$ is the sum of $\neg_{ker}$ and $\neg_{supp}$. As expected, the fact that it acts both on the kernel and the support makes it outperform its parts in most cases.

Composition

We compare four different composition operations; *spider, phaser, fuzz* and *diag*. As expected, *diag* does not give statistically relevant correlations. It puts the result in the computation basis, erasing both inputs' eigenbasis and removing relevant information. *Fuzz* does not perform well either, with only one correlation above 0.4 (in the basis of the word, marked by 'w', and in combination with $\neg_{sub}$ and sim_{trace}).

The best operation is *phaser*, which not only gives the maximal correlation but performs well with most similarity measures — in the basis 'w' and with $\neg_{sub}$ it gives a correlation above 0.29 for all similarity measures. Even with $\neg_{inv}$ and $\neg_{ker}$ and in the basis 'w', it has correlations of at least 0.18. In the basis of 'w' it only does not interact well with $\neg_{supp}$. Overall, this speaks to a desirable robustness of the results provided by the *phaser*.

The *spider* also gives good correlations, though not as high as the *phaser*. It is the only composition operation that performs better in the basis of the context ('c') than in the basis of the word ('w'). This makes the *spider* also the only composition operation where the choice between framework CN_{word1} and CN_{word2} becomes truly relevant, giving correlations above 0.45 only for the first framework.

Basis

Spider, fuzz and *phaser* are basis dependent. The choice of basis of the composition determines the eigenbasis of the result. In our experiments, we compare choosing between one of two bases. The first basis is the eigenbasis of the word being negated, to which we refer as 'w'. The second basis is the eigenbasis of the context to which we refer as 'c'. Choosing between those two bases corresponds to choosing which word determines the eigenbasis of the outcome while the other word informs the spectrum. For *fuzz* and *phaser* 'w' (the basis of the negated word) performs better. This finding matches our psychological motivation of updating the results of the logical negation by the context. The application of the context updates the eigenspectrum of the logical negation; we leverage the worldly knowledge to grade the weights of the logical negation.

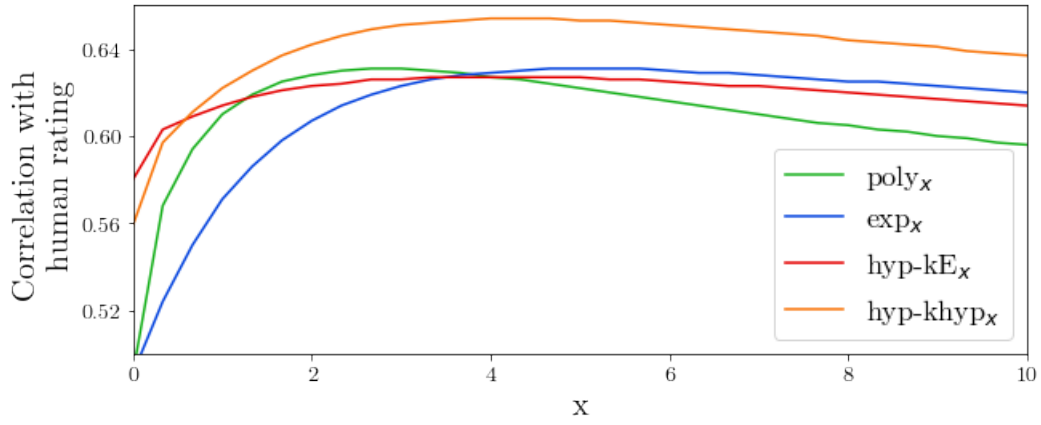


Figure 4.2: Influence of different context functions on correlation of CN_{word1} (with $phaser_c$, $\neg_{sub}$ and sim_{trace}) with human intuition

For the *spider*, the eigenbasis of the context 'c' gives better correlations. Here it is important to observe that CN_{word1} and CN_{word2} have different correlations when choosing the basis of the context. CN_{word1} outperforms CN_{word2} in the basis of the context. Intuitively, it seems better to consider the basis of the individual words rather than the combination of the words (i.e. the worldly context).

4.5.3.3 Context comparison

The context functions we explore are:

$$p_{h_i} = \texttt{poly}_x(i) \coloneqq (n-i)^x$$
$$p_{h_i} = \texttt{exp}_x(i) \coloneqq (1+\frac{x}{10})^{(n-i)}$$
$$p_{h_i} = \texttt{hyp-kE}_x(i) \coloneqq (n-i)^{\frac{x}{2}} k_{\mathsf{E}}(w, h_i)$$
$$p_{h_i} = \texttt{hyp-khyp}_x(i) \coloneqq (n-i)^{\frac{x}{2}} k_{\mathsf{hyp}}(h_i, w)$$

for a word w with the hypernyms $h_1, ..., h_n$ ordered from closest to furthest.

Figure 4.2 shows the correlation with human rating (on the y-axis) in relation to the x parameter of the four correlation functions. We recall that the x parameter quantifies to what degree the distance in the entailment hierarchy should be considered. To obtain these values, we used the highest performing negation framework (CN_{word1}/CN_{word2} with the *phaser* in the basis 'w' and the sim_{trace}).

We can see that the first three context functions peak around 0.630. $\texttt{hyp-khyp}_x$ peaks at a maximal correlation of 0.654, therefore slightly outperforming the other context functions. All four functions eventually peak, showing that the context should not be too close to the original word. Additionally we can observe that at $x = 0$, $\texttt{hyp-kE}_0(i) = k_{\mathsf{E}}(w, h_i)$ and $\texttt{hyp-khyp}_0(i) = k_{\mathsf{hyp}}(w, h_i)$ still perform very well. Both functions achieve correlations above 0.56 with $\texttt{hyp-kE}_0$ scoring as high as 0.58. As at $x = 0$, these functions do not take the WordNet distance into account. This observation indicates the potential of the proposal in Section 4.3.2. However, it is important to observe that WordNet still informs the words being explored as context. So these context functions only remove the dependence on WordNet for determining the weights.

The frameworks CN_{word3} and CN_{word4} perform slightly better under the polynomial context function. However, the basic observations are comparable. Thus we omit their graphs here. As they perform optimally under $\texttt{poly}_4$, we have added a second framework comparison table in the Appendix, which is constructed with that context function (see Appendix B.1).

4.5.3.4 Similarity measure comparison

The best correlations are generally achieved with the trace similarity and k_{E2}. The fact that the trace similarity does well is expected as similar similarity measures for vectors have achieved high correlations in the original experiments (Kruszewski et al., 2016). While trace similarity might be a high performing similarity measure, it is not useful as an entailment measure since it is symmetric. The second-best performing similarity measure is k_{E2}. Under $\texttt{poly}_4$, k_{E2} scores as high as 0.604 (see Appendix B.1). The good performance of k_{E} is surprising when considering that the theoretic analysis of its interaction gives unfavourable results (see Table 4.1).

k_{hyp1} and k_{BA} are the only similarity measures that give reliably positive correlation (ranging from 0.102 to 0.312 for k_{hyp1} and 0.106 to 0.303 for k_{BA}). All other measures are more volatile with respect to the composition operation, basis or logical negation.

For the asymmetric measures, the comparison from w_A to $CN_{word}(w_N)$ mostly performs better than from $CN_{word}(w_N)$ to w_A, i.e. k_{E2} outperforms k_{E1}. This holds for k_{E} and k_{hyp} but also for k_{BA} which is symmetric up to a factor of -1. For k_{BA} too w_A to $CN_{word}(w_N)$ gives the positive correlations, while the other direction gives negative correlations. This is rather surprising, as the question "`How much does not` w_N `entail` w_A`?`" seems more relevant than "`How much does` w_A `entail not` w_N`?`". However, the latter seems to give results closer to human intuition.

4.6 Additional exploration

For the previous experiments, it is crucial to observe that any conversational negation operation must perform well in these experiments. However, a good performance is not sufficient for something to be a negation operation. A good example of this are leaving the word as is, i.e. the identity operation, and taking the worldly context. They are both obviously not good choices for conversational negation. Nevertheless, they perform pretty well in our experiments.

We have four different frameworks that perform well in the experiments. Each of them is a potential candidate for being the operation for conversational negation of choice. However, further explorations are necessary to evaluate their performance.

Intuitively, we would like the result of the negation to be as different from the original as possible while still observing our previous experiments. The results of logical negation, in a sense, form the complete opposite of the original, but they defy human intuition and thus do not pass our experiments. To get an idea how the outcomes of our frameworks compare, we compare the following six operations; the identity (marked by 'w_N'), taking the worldly context (marked by '$\mathtt{wc}_{w_N}$'), logical negation (marked by $\neg$), the CN_{word1} (with the identity negation and $phaser_w$ - this is equal to CN_{word2} under the same configuration.), the CN_{word3} and CN_{word4}. For each of these six operations, we calculate the negation of w_N under the operations and pair-wise compare the results. For this comparison, we take the Frobenius norm of the difference between the two resulting matrices. The averages of the pair-wise similarities are shown in Table 4.3. To get a more intuitive feeling for

Table 4.3: Average distance between various negation operations (Frobenius norm of the difference of any two operations)

	w_N	$\texttt{WC}w_N$	$\neg$	CN_{word1}	CN_{word3}	CN_{word4}
w_N	0.0	0.655	6.816	0.686	0.998	1.073
$\texttt{WC}w_N$	0.655	0.0	6.542	0.262	0.744	0.708
$\neg$	6.816	6.542	0.0	6.548	6.603	6.548
CN_{word1}	0.686	0.262	6.548	0.0	0.713	0.693
CN_{word3}	0.998	0.744	6.603	0.713	0.0	0.156
CN_{word4}	1.073	0.708	6.548	0.693	0.156	0.0

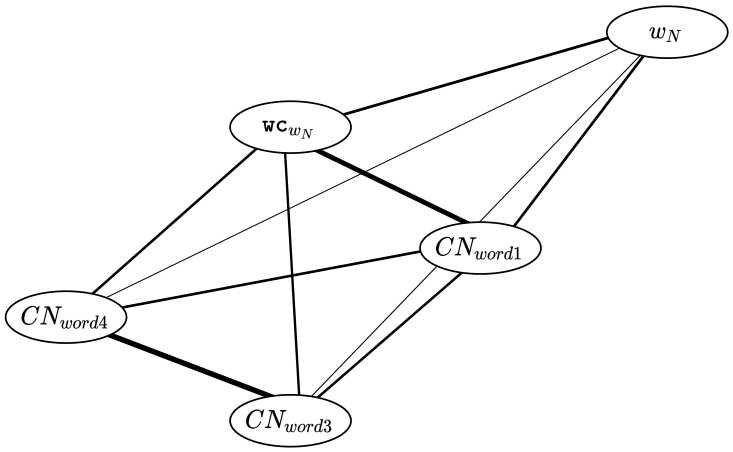


Figure 4.3: Visualisation of distance between negation operations. The edge weight and length is informed by their average distance (displayed in Table 4.3)

the results, we visualise them in Figure 4.3. We use the Fruchterman-Reingold force-directed algorithm (Hansen et al., 2020) to draw the results as a weighted graph. Each operation corresponds to one vertex, and their similarity weights the edges. Their weight informs the edge width and length. We omitted the logical negation as it would destroy the nuances of the other relations due to its high weights. We can see that the word, w_N, and worldly context, $\texttt{WC}w_N$, are quite close. Surprisingly, CN_{word1} also seems to produce results that are close to these two vertices. Intuitively, this does not speak for the framework to be a good choice of negation as we would expect the negation of a word to be dissimilar to the original. The other two frameworks are much further away.

Further explorations of the CN_{word1} and CN_{word2} show that the impact of the logical negation is less than initially expected. Words, represented by 50×50 matrices, often have one or two larger eigenvalues, with the rest being near 0.

Thus the logical negation of such a word results in a positive operator with many eigenvalues close to 1. This result is often similar to the identity matrix. After the logical negation, the impact of the original word on the overall outcome is smaller than expected. This lets us conclude that the current implementation of CN_{word1} might not be an optimal candidate for conversational negation. To be precise, the fault most likely lies with our choice of logical negation as it seems, for our dataset, to make all outcomes similar to the identity matrix, independent on the input.

Based on these observations, we can conclude that additional experiments are necessary to validate our proposals. Hence, new experimental designs have to be conceived, and data sets created. This is left for future work. However, we have shown that our frameworks are decent candidates, all built based on the same hypothesis; negation is context dependent.

Additionally, it is unclear if our current method to build the positive operators is optimal. We build our positive operators from vectors created via co-occurrence. As a word and its negation often appear in similar contexts (Kruszewski et al., 2016), the outcome of negation must be similar to the original. This complicates the process of negation, as the correct result is not intuitively apparent. Other methods to create positive operators or other meaning representations, such as conceptual spaces, might overcome these complications. The proposed frameworks, mainly the first two, are valid in all meaning spaces that offer weighted mixtures and logical negations. Therefore, they could be applied to other meaning representations.

Scaling up

5

Conversational negation of multiple words

Contents

5.1 Intuition

In this section, we will take the negation of words to the negation of sentences. As we will be relying on the negation of individual words, we require the constituents of our sentence to be meaning states, similar to sentences in the DisCoCat framework. We do not yet allow for evolving meanings, i.e. open wires. Still, this section will be written for the DisCoCirc framework. Therefore, we will view sentences as meaning states updating wires. This will later allow us to combine the negation of multiple words with the negation of wires to construct one all-encompassing negation for the DisCoCirc framework. For now, we will assume that all inputs to a sentence are states and can therefore be negated by the negation operation proposed in the previous section.

As pointed out by Oaksford and Stenning (1992), under the alternatives view, the negation of a sentence can be viewed as a negation of a subset of its words. The negation of "`Bob drove to Oxford by car.`" could be interpreted as:

a) `Bob did not drive to Oxford by car - Alice did`

b) `Bob did not drive to Oxford by car - He carpooled`

c) `Bob did not drive to Oxford by car - He drove to Cambridge`

d) `Bob did not drive to Oxford by car - He drove a van`

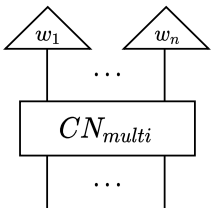
e) `Bob did not drive to Oxford by car - Alice carpooled to Oxford`

where the underlined word(s) are the target of the negation. The last sentence is one of many examples of multiple constituents being negated at once. Thus the interpretation of the negation depends on not only the elicited alternatives but also the target of the negation. While some of these interpretations seem to be more plausible than others, the listener has to use different sources of context to derive which interpretation is correct.

In contrast to the negation of words, we observe that the search for alternatives does not always occur during the negation of sentences. As pointed out by Prado and Noveck (2006), negation can also be used for simple information denial. For example, let us consider the sentence "`I have not watched Pulp Fiction.`". While this could be interpreted based on the previous intuition (maybe someone else watched it), the statement could also convey information denial. In a conversation about movies, I may simply want to express that I have not watched this particular movie yet. While information denial could potentially be modelled via logical negation — as assumed in Chapter 4 for individual words — our experimental validation does not tackle this challenge. In particular, we do not know how logical negation acts when negating multiple words instead of just a single word as initially proposed by Lewis (2020). Therefore, for the remainder of this thesis, we will continue to consider negation under the *search-for-alternatives-view* and leave the *narrow view* for future work.

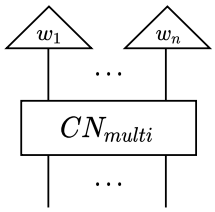
5.2 The framework

Based on the previous intuition, we view the negation of multiple words as a negation of a subset of the words. For this, we propose to utilise the previously introduced framework to model the conversational negation of individual words (see Section 4). As the correct target of the negation is not usually obvious, we once more utilise a weighted sum. We sum over the different interpretations of the negation, depending on the intended target. Thus the negation of the n words $w_1, ..., w_n$ is a mixture of all interpretations where we negate only one of the words plus all the negations where we negate two words and so on:



$$\begin{aligned} := \quad & \sum_{i=1}^{n} p_{\{w_i\}} \left(w_1 \otimes \cdots \otimes CN_{word}(w_i) \otimes \cdots \otimes w_n\right) \; + \\ & \sum_{i=1}^{n} \sum_{j=i+1}^{n} p_{\{w_i, w_j\}} \left(w_1 \otimes \cdots \otimes CN_{word}(w_i) \otimes \ldots \right. \\ & \qquad \left. \otimes CN_{word}(w_j) \otimes \cdots \otimes w_n\right) \\ & + \quad \ldots \end{aligned}$$

More formally for some set of words $W = \{w_1, ..., w_n\}$, we define the negation to be the sum over all non-empty subsets $W' \subseteq W$, called the *negation sets*, to get:



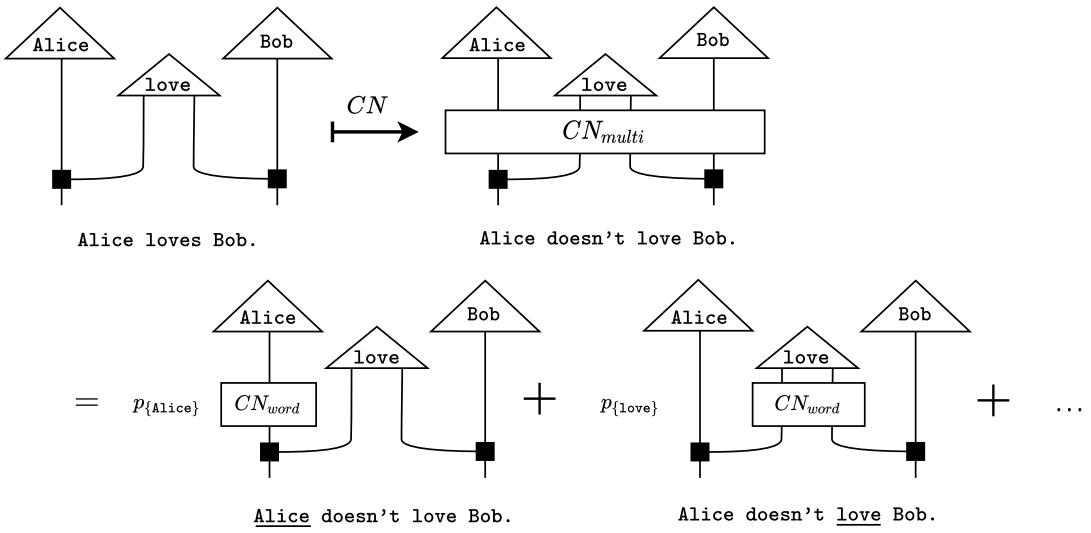
$$:= \quad \sum_{\substack{W' \subseteq W \\ W' \neq \emptyset}} p_{W'} \bigotimes_{i=1}^{n} \begin{cases} w_i & \text{if } w_i \notin W' \\ CN_{word}(w_i) & \text{if } w_i \in W' \end{cases}$$

Each summand of the weighted sum, identified by a negation set, thus corresponds to one set of words targeted by the negation. Each negation set corresponds to a different interpretation. We enforce the negation set to be non-empty, as an empty negation set would correspond to never applying CN_{word}. We would thus get the original, positive meaning as an interpretation of our negation.

Sentences in language circuits are made up of two parts; meaning states and updates. The updates combine the evolving meanings present in the text and the

meaning states. The conversational negation of multiple words manipulates the value of the meaning states but leaves the structure of the updates untouched. Thus, the negation of multiple words can be computed by adding the negation operation before the meaning updates.

We propose to view negation as a function that acts upon the language circuit of a sentence without a negation. We call this function CN for conversational negation. Thus, for example, applying this function to the sentence "`Alice loves Bob`" gives us:



where we sum over all non-empty subsets of {`Alice, love, Bob`}. The alternatives that are being elicited for the negated words are determined by the CN_{word} operation. Here we can use one of the frameworks proposed in the previous chapter. Alternatively, CN_{multi} works with any unitary conversational negation operation on meaning states that models conversational negation.

5.3 Determining the context

For the conversational negation of multiple words, the context informs the targets of the negation. Each selection of targets to be negated then correspond to one interpretation of the negation. The remaining challenge is to extract the appropriate weights for each interpretation from the context. The context can

come in various forms, such as the person who is speaking and their intention. In spoken language, intonation can inform the correct interpretation by the speaker emphasising the negation's target.

For more complex sentences, the grammatical structure can be another factor. Take the previous example; "`Bob did not drive to Oxford by car`". This sentence adds the final detail of the mode of transport — `by car`. Intuitively, such a detail is more likely to be the intention of the negation. If the speaker would simply like to express that Bob drove to a different location, the shorter sentence "`Bob did not drive to Oxford`" would suffice. The second example, "`Alice does not love Bob`", is more ambiguous with respect to the grammatical structure.

Another source of context is the surrounding text; the interpretation of the negation should line up with the information conveyed in the remaining text. Let us, for example, consider a text about `Bob`'s favourite car trip. In that context, the target of the negation "`Bob did not drive to Oxford by car`" is probably the location of the trip.

Overall no single source of context is sufficient. For optimal results, various mixtures of context are required. However, we can make a general observation; psychologically, we know that humans are less capable of focusing on a large number of details. This also holds for negations (Evans, 1989; Oaksford & Stenning, 1992). Thus we can say that larger negation sets should generally have lower weights than smaller negation sets. In fact, Oaksford and Stenning (1992) only consider negation sets of size 1. Intuitively this corresponds to the alternative "`Alice drove to London.`" to be perceived as less plausible when considering "`Bob did not drive to Oxford by car.`". Such an interpretation would require considerably more contextual pressure to be reasonable.

In contrast to the conversational negation of words, we do not rely on the DisCoCirc framework to extract the context from the text. Instead, we propose to derive the influence of the context externally. This is due to the higher complexity of the negation, which current encodings of ambiguity do not capture.

5.3.1 Weights from entailment - an example

This section will give an intuition on how to quantify the context from surrounding sentences using entailment measures. Interpretations of a negation that highly entail the surrounding text are more likely to be the intended meaning. They should thus get a higher weight. For this quantification, sentences closer to the negation should have a greater influence than sentences further away.

To illustrate this intuition, let us consider a simplified example where we have a negation followed immediately by a clarification. Both sentences are of the identical, simple grammatical structure:

`This is not a cute dog`

`This is a cute cat`

We color the sentences to ease the reading of this example with the negated sentence being red. For the sake of simplicity, we will assume that the possible negation sets are {`cute`}, {`dog`} and {`cute, dog`}, ignoring that `this` could also be part of the intended target[1]. We thus have to determine the respective weights $p_{\{\texttt{cute}\}}, p_{\{\texttt{dog}\}}$ and $p_{\{\texttt{cute, dog}\}}$. To the human reader, given the clarification in the second sentence, the correct interpretation of the negation is obvious. The object we are talking about seems to be a "`cute cat`", which is something `cute` that is not a `dog`. Therefore the correct negation set should be {`dog`}.

To arrive at the same result, we will compare each possible interpretation of the negation with the follow-up sentence. For this, we will use the previously introduced entailment measures, represented by $\sqsubseteq$. While we have performed small scale experiments upon which we base our intuitions, more extensive experiments have to be conceived to draw reliable conclusions. For now, we will rely on intuitions to convey our ideas. Our entailments will be categorised into *low, medium, high* and *fully*, where *fully* corresponds to maximal entailment, reserved for the entailment of a word with itself. We use the fact that our two sentences are of the same

[1]The alternative elicited by the negation could be "`This is not a cute dog - that is`". We will ignore this case to reduce the number of possible negation sets.

grammatical structure and compare them word-by-word, i.e. we compare the adjectives and nouns, respectively. We have:

- not cute dog $\sqsubseteq$ cute cat - cute *fully* entails cute, as something cute is indeed cute. The entailment form CN_{word}(dog) to cat is *medium*. Something that is not a dog could be many other animals including a cat. Overall the entailment is *high*. We have:

not cute dog	cute cat	Entailment
cute	cute	*fully*
CN_{word}(dog)	cat	*medium*
Overall:		*high*

- not cute dog $\sqsubseteq$ cute cat - The entailment of CN_{word}(cute) with cute is *medium*. This is due to the fact that in distributional semantics, negation of a word and the word itself appear in similar contexts (Mohammad et al., 2013; Oaksford & Stenning, 1992). However, the entailment from dog to cat is *low*; something that is a dog is not a cat. Therefore the overall score of this interpretation is *low*.

not cute dog	cute cat	Entailment
CN_{word}(cute)	cute	*medium*
dog	cat	*low*
Overall:		*low*

- not cute dog $\sqsubseteq$ cute cat - CN_{word}(cute) and cute have a *medium* entailment due to them appearing in similar contexts. Similarly the entailment from CN_{word}(dog) to cat is *medium*. Overall the entailment in *medium*.

not cute dog	cute cat	Entailment
CN_{word}(cute)	cute	*medium*
CN_{word}(dog)	cat	*medium*
Overall:		*medium*

Thus the negation set {dog} has the highest entailment, matching our intuition as humans.

In this example, we rely on both sentences having the same grammatical structure to compare the sentences word-by-word. One goal would be to compare any two sentences, independent of their grammatical structure. This would, amongst other things, require an update mechanism that preserves entailment relations during composition. This is open research. Some promising results can be found in De las Cuevas et al. (2020), Kartsaklis and Sadrzadeh (2016a, 2016b), and Sadrzadeh et al. (2018).

Opening the wires

6 Conversational negation of evolving meaning

Contents

6.1 Intuition

One central feature of the DisCoCirc framework is that it allows for meanings of dynamic words to evolve throughout a text. Meanings of dynamic words, such as actors in a story, become wires instead of simple states. This change allows for the text to update these words as the story evolves. When designing an operation for conversational negation applicable in DisCoCirc, these evolving meanings need to be treated differently than static words. We will therefore propose a new framework for the negation of evolving meanings.

Let us consider a text with `Alice, Bob, Charles` and `Dave`. These four actors

will be our dynamic words. Let us assume we know the following details about them:

`Alice is a human.`	`Alice is a mathematician.`
`Bob is a human.`	`Bob is a physicist.`
`Charles is a human.`	`Charles is a pianist.`
`Dave is a dog.`	`Dave is a pet.`

Let us now consider the following sentence:

`Alice does not publish a paper.`

To interpret this negation, the first task would be to determine the correct negation set. The target of the negation could contain `Alice, publish` or the `paper`. Let us assume we know that a paper is being published. Thus we know the intention of this negation is to express that someone other than `Alice` publishes a paper. To model this sentence, we have to model the negation of the evolving meaning, which we call `Alice`.

Intuitively, if `Alice` is not publishing the paper, someone else is. This someone else must either be `Bob, Charles` or `Dave`. Thus to model the interpretation "`Alice does not publish a paper.`", we have to consider these three alternatives. Knowing the information we previously gained about our actors, a human reader would agree that the most likely alternative is `Bob` - another scientist. However, `Charles`, being another human, is still more likely to publish a paper than the dog `Dave`.

We observe that under this interpretation, we do not learn anything about `Alice`. We only gain information about the other actors in the text. If the intention of the negation were to give us more information about `Alice`, we would have picked a different negation set. For example the negation set {`publish`} could be interpreted as:

`Alice does not publish a paper - She is writing on a paper`

The intended meaning of the negation is most likely to be a mixture of these interpretations, for which we will again use ambiguity via weighted sums. However,

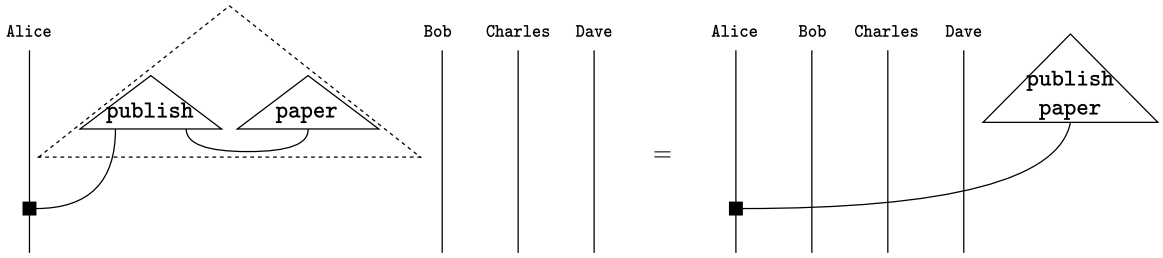
in this section, we will model negation sets containing only dynamic words, such as, in this particular case, the negation set {`Alice`}.

6.2 The framework

To develop a framework for this negation, we will first consider the positive sentence:

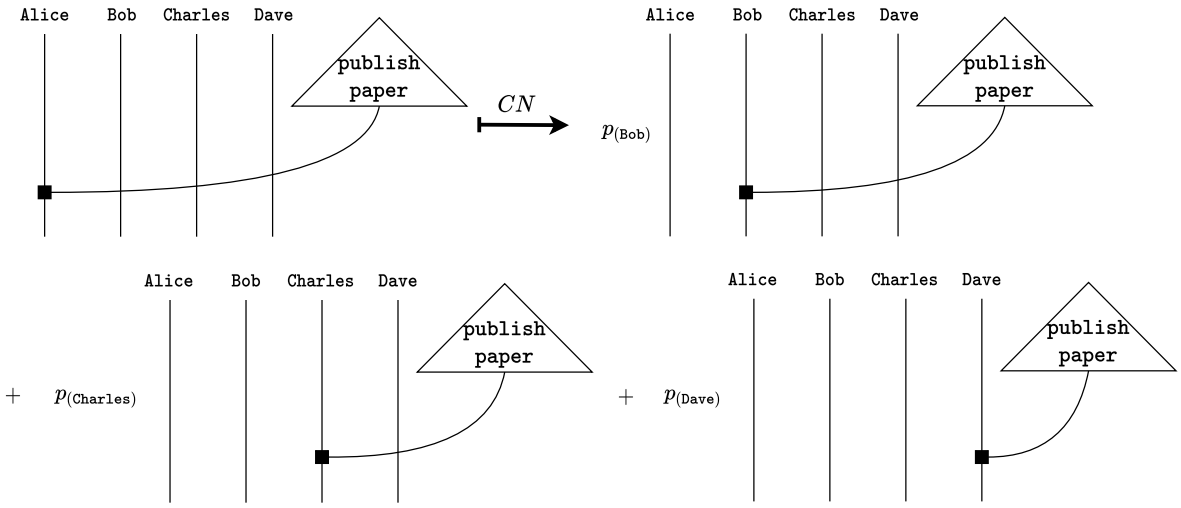
`Alice publishes a paper.`

Diagrammatically the sentence would look like this:



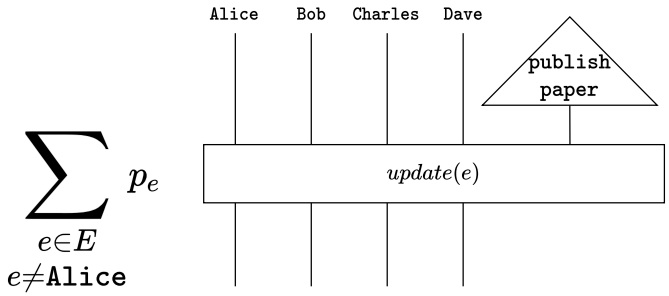
where on the right-hand side, we merge "`publishes a paper`" into a single meaning state as indicated by the dotted triangle on the left-hand side. Additionally, we use the symmetry of our category to move this state to the right of all actors. This second step will later simplify our generalisations.

To model this interpretation of the negation, we observe that the meaning of the negated sentence becomes "`Someone other than Alice publishes a paper.`". While we may not definitively know who that other person is, we assume it must be another person in our story. Thus it must be one of the other evolving meanings. Applying the negation function to our sentence, we thus get:

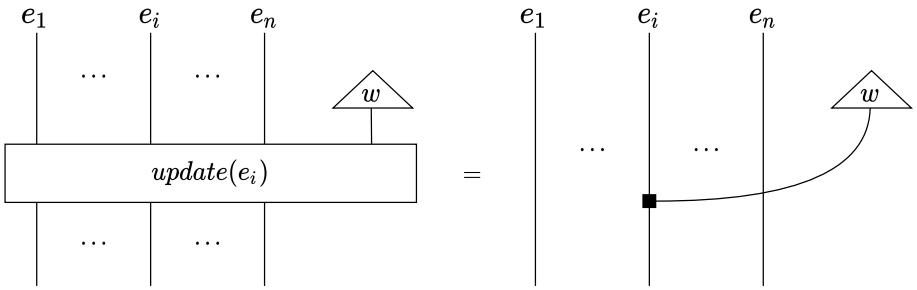


The negation becomes a weighted sum of the actors other than `Alice` having published a paper. The weights $p_{(\texttt{Bob})}, p_{(\texttt{Charles})}$ and $p_{(\texttt{Dave})}$ correspond to the plausibility of these respective evolving meanings to be the correct interpretation.

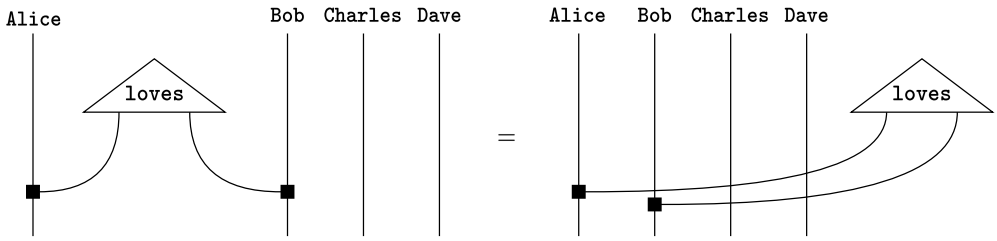
More formally we have:



with the set $E = \{\texttt{Alice}, \texttt{Bob}, \texttt{Charles}, \texttt{Dave}\}$ being our evolving meanings. $update(e_i)$ corresponds to updating the i-th evolving meaning with the meaning state:



As a second example, we will consider another sentence, which updates multiple evolving meanings at once. Let us consider "`Alice loves Bob.`". This sentence has two updates; one to `Alice` and one to `Bob`. If we take the same evolving meanings as in the previous example, this sentence will look like this:

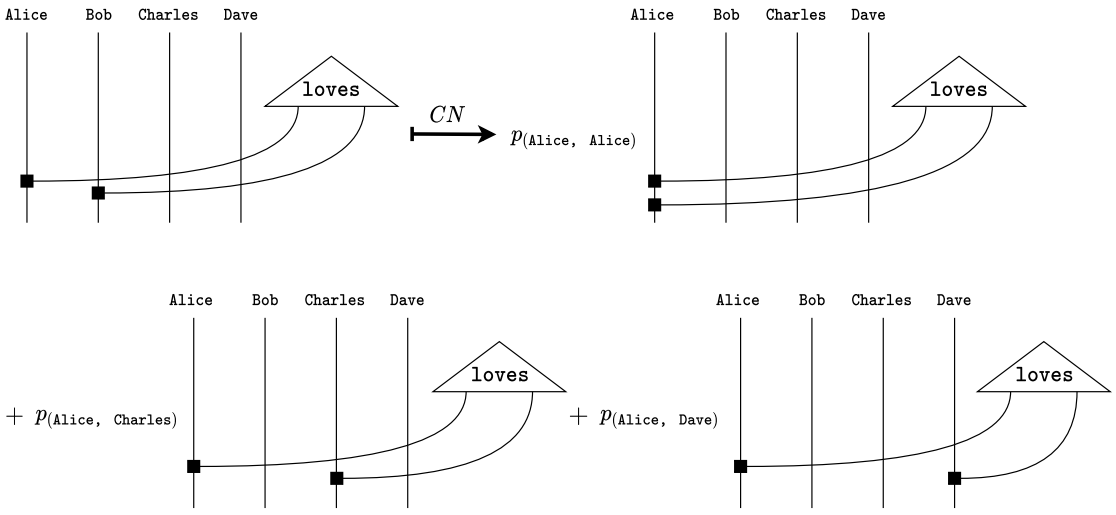


On the right-hand side, we once more move the meaning state to the right using the symmetry.

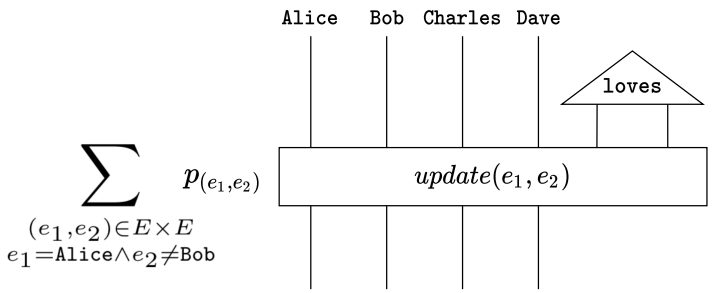
Let us assume we wanted to model the negation of this sentence, in particular the negation set {Bob}. Thus we want to model the sentence "Alice loves someone other than Bob". In our story that could be Charles or Dave but it could also be Alice herself. This last alternative can be read as:

Alice does not love Bob - She only loves herself

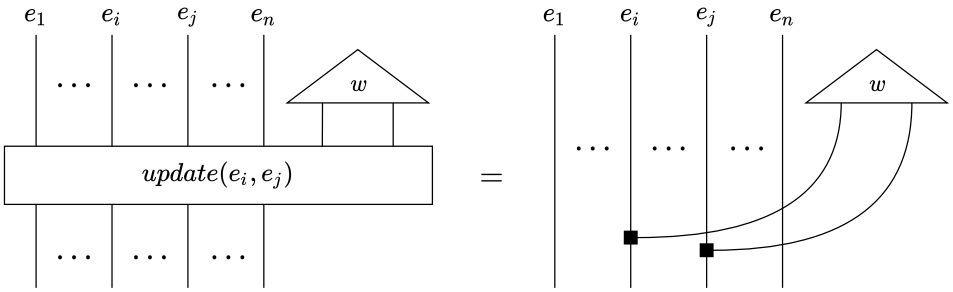
Diagrammatically these three cases would thus correspond to:



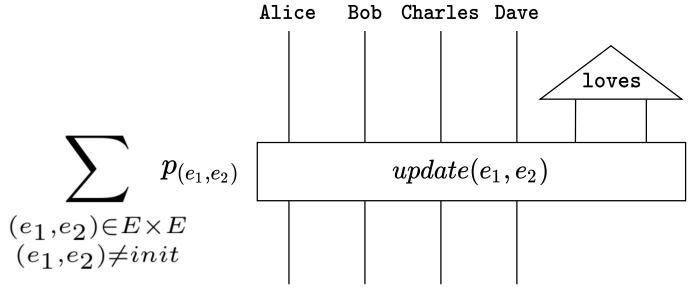
More formally, the update function now contains two updates. To interpret our negation, we move the second update box around while leaving the first one in place. Thus we can write our negation as:



where $E = \{\texttt{Alice, Bob, Charles, Dave}\}$ is the set of our evolving meanings. We define the first update to be on `Alice` and the second update on someone other than `Bob`. The update box is defined as:

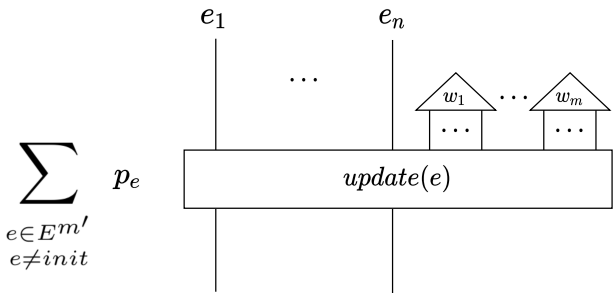


We can generalise the negation to cover all interpretations where we negate a non-empty subset of the evolving meanings. Thus instead of only modeling "`Alice does not love Bob`." we model the negation sets {`Alice`}, {`Bob`} and {`Alice, Bob`} simultaneously with one sum. We have:



with $init = (\texttt{Alice, Bob}) \in E \times E$ is the initial update structure on the positive sentence. We thus sum over all update structures which are not the original one. Each of our three negation sets contains multiple of the summands. {`Alice`} contains all updates where $e_1 \neq \texttt{Alice} \wedge e_2 = \texttt{Bob}$, i.e. all updates where only `Alice` is negated. {`Bob`} contains all updates where $e_1 = \texttt{Alice} \wedge e_2 \neq \texttt{Bob}$ and finally {`Alice, Bob`} contains the remaining updates for which we have $e_1 \neq \texttt{Alice} \wedge e_2 \neq \texttt{Bob}$. Thus the sum covers exactly all non-empty negation sets of the evolving meanings. Once more, there is a grading in the plausibility of each interpretation enforced by the respective weights $p_{(e_1,e_2)}$.

In general, we can take any sentence with m words and a total of m' updates in a text of n evolving meanings. The negation of any subset of the evolving meanings could be modelled as follows:



where *init* is the original update structure of the sentence and $update(e)$ performs m' updates specified by $e \in E^{m'}$. We sum over all update structures that are not the original one.

The conversational negation of evolving meanings can be viewed as logical negation on the set of evolving meanings. After the logical negation, it is graded by the weights to align the results with human intuition. The negation of evolving meanings changes the update structure but leaves the meaning states untouched. This directly contrasts with the negation of multiple words, which changes the meaning states but leaves the update structure in place.

We are currently not modelling the option that the alternative to the negated dynamic word is not present in the circuit. Maybe `Alice` loves neither herself, `Bob, Charles` nor `Dave` but some other person. One option is for each negation to introduce a new wire, which represents other potential lovers of `Alice`. However, such a design choice would raise many other questions; is that wire ever accessible for future updates? What if we do figure out that `Alice` loves `Dave`? Then the wire becomes obsolete. We leave this question to future work and assume that all possible alternative evolving meanings are present in the text and, therefore, in the circuit. This assumption is reasonable as the intention of the evolving meanings is to capture all entities about which we might want to talk. Therefore they should also include the potential alternatives to a negation.

6.3 Determining the context

After having proposed a framework to model the negation of evolving meanings, the remaining challenge becomes once again to derive the weights that inform

our interpretations.

Two primary sources inform our weights; similarity and relations. Similarity refers to evolving meanings having similar attributes. In our previous example, the most plausible alternative to the one scientist is the other scientist. This intuition is based on the knowledge we have about the attributes of the actors. The second source, relations, refers to evolving meanings having a connection. For example, we could imagine two siblings that are in no way similar - different preferences, jobs, gender. However, the sentence "`Sibling`$_{\texttt{A}}$ `does not take care of their sick mother.`" could plausibly be interpreted as the other sibling taking care of the mother instead. Despite them not being similar, having a relation — being siblings — makes them plausible alternatives to one another.

The second source of context could potentially be modelled with relation graphs extracted from the text. This is left for future work. For the first source of context, we will propose an approach to quantify the similarity of evolving meanings via the previously introduced negation of multiple words (see Section 5).

6.3.1 Weights from similarity - actors as sentences

6.3.1.1 Intuition

One key feature of language circuits is removing grammatical complexity and extracting the core meaning updates of sentences. This makes different texts with the same informational content identical. For example, the two sentences:

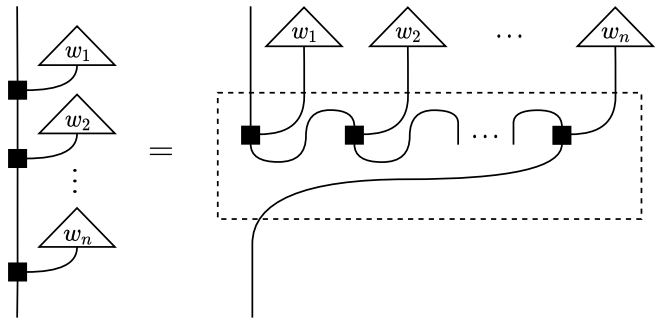
```
Bob is a dog.
Bob is happy.
```

result in the same circuit as the single sentence:

```
Bob, who is a dog, is happy.
```

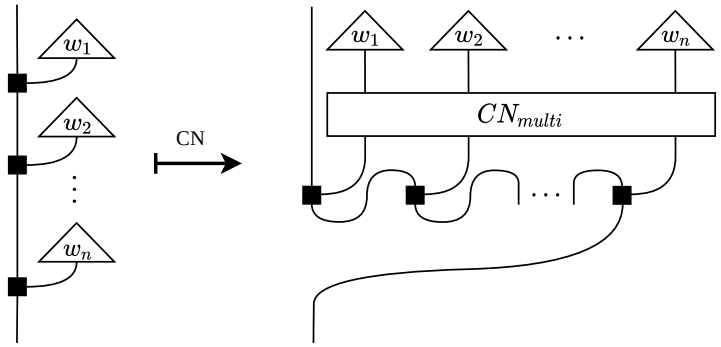
We can observe this feature to motivate viewing the updates on an actor as a single long sentence of static words. A series of updates to a wire can be bent using the yanking equations. The series of updates can then be seen as a single

process through which multiple meaning updates inform a single wire. For example, for the meaning updates $w_1, ..., w_n$ on some wire we have:



where on the right-hand side, the dotted box surrounds a single process that updates the incoming wire.

We can now use the negation of multiple words on the meaning states updating the wire. We thus get:



The outcome of this process then carries the meaning of things being similar but not equal to the original.

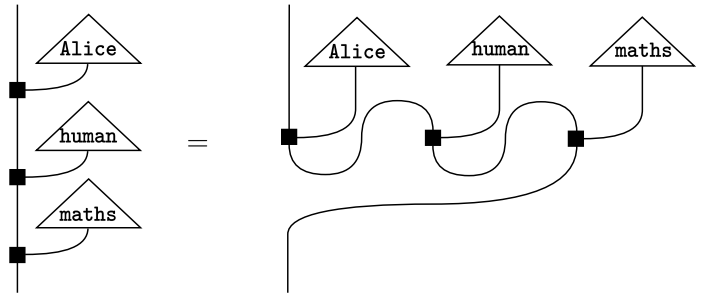
6.3.1.2 An example

To explore an example, we can apply this to our previous story where "`Alice does not publish a paper`" to find plausible alternatives to `Alice`. We recall that our text has the following four actors:

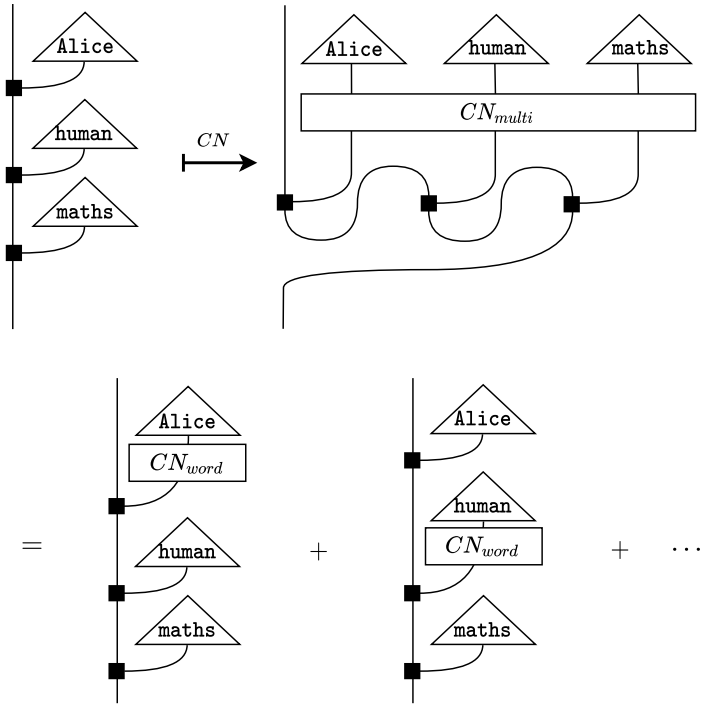
`Alice is a human.`	`Alice is a mathematician.`
`Bob is a human.`	`Bob is a physicist.`
`Charles is a human.`	`Charles is a pianist.`
`Dave is a dog.`	`Dave is a pet.`

`Alice`'s wire has three updates; the fact that it carries the meaning of something called `Alice`, the fact that this is a `human` and a `scientist`. The first meaning update with `Alice` contains all preconceptions about something that is an `Alice` which are then refined with the further meaning updates.

Applying the yanking equations to `Alice`'s wire, we get:



Applying the negation function then gives us:



Therefore the negated wire of `Alice`, to which we will refer as "not `Alice`", is a weighted sum over the different negation sets containing {`Alice, human, mathematician`}. "not `Alice`" might thus be someone who is a `human` and a `mathematician` but not called `Alice` (negation set {`Alice`}). Or "not `Alice`" refers to another person with the same name who is not a `mathematician` (negation set {`mathematician`}). Which negation set, and thus interpretation of the negation,

is correct depends on the context. Here methods such as proposed in Section 5.3 could be applied.

The possible alternatives to `Alice`, namely `Bob, Charles` and `Dave` have different entailment from "not `Alice`". This entailment value depends on the choice of negation set. To see what the maximal entailment from "not `Alice`" could be, we will pick the best negation set for each alternative. Similar to the previous example in Section 5.3.1 we will refrain from actual numbers and instead give intuitions on the level of entailment. Once more, these will be categorised into *low, medium, high* and *fully*, where *fully* corresponds to maximal entailment from of a word to itself.

To simplify this example, we will use the fact that each evolving meaning has three attributes; their name, their genus and their profession. We will thus compare these attributes one-by-one. In an actual text, this simplification is unrealistic. Instead, we should compare the actual state of the evolving meanings, relying on the update mechanisms to preserve entailment. This is a similar observation as made in Section 5.3 and is equally reliant on future work. In our simplified scenario, we have:

- `Bob` is a human physicist. Thus the highest entailment between "not `Alice`" and `Bob` can be achieved, when picking the negation set {`Alice`, `mathematician`}. We will compare the values of `Bob` and "not `Alice`" under this particular negation set element-wise to gain an intuition on the overall entailment.

  We have that someone who is not named `Alice` can have many other names including `Bob`. Thus the entailment from CN_{word}(`Alice`) to `Bob` is *medium*. `human` fully entails itself. CN_{word}(`mathematician`) *highly* entails `physicist`, as physics is a very plausible alternative to mathematics. Overall we get a *high* entailment. We have:

"not `Alice`"	`Bob`	Entailment
CN_{word}(`Alice`)	`Bob`	*medium*
`human`	`human`	*fully*
CN_{word}(`mathematician`)	`physicist`	*high*
Overall:		*high*

- `Claire` is a human pianist. As for `Bob` the best entailment can be achieved with the negation set {`Alice`, `mathematician`}. We will once more consider the entailments between "not `Alice`" and `Claire`.

  Similar to `Bob`, we have that CN_{word}(`Alice`) entails `Claire` with *medium* entailment and `human` *fully* entails itself. However, CN_{word}(`mathematician`) entails `pianist` with only *medium* entailment. While both of them are professions, we are less likely to think of a `pianist` when saying that someone is not a `mathematician`. Overall the entailment is *medium*.

"not `Alice`"	`Claire`	Entailment
CN_{word}(`Alice`)	`Claire`	*medium*
`human`	`human`	*fully*
CN_{word}(`mathematician`)	`pianist`	*medium*
Overall:		*medium*

- `Dave` is a pet dog. For him the best negation set is {`Alice`, `human`, `mathematician`}.

  We have that CN_{word}(`Alice`) entails `Dave` with medium entailment. Our preconceptions about the name `Dave` are as similar to our preconceptions about `Alice` as they are for `Bob` or `Claire`. Something that is not `human` could be a `dog` with *medium* entailment. But we do not think of a `pet` when talking about not `mathematician`. Thus this final entailment is *low*. Overall we have a *low* entailment. We have:

"not `Alice`"	`Dave`	Entailment
CN_{word}(`Alice`)	`Dave`	*medium*
CN_{word}(`human`)	`dog`	*medium*
CN_{word}(`mathematician`)	`pet`	*low*
Overall:		*low*

Overall `Bob` has the highest potential entailment from `Alice`. `Charles` has a higher potential than `Dave`. This matches our intuition of `Bob` being a reasonable alternative. However, the final outcome is dependent on the context, which determines the

negation set. Thus, if the negation set is {`Alice, human, mathematician`}, `Dave` might still have a higher entailment than the other two alternatives.

This example gives an intuition on how conversational negation of multiple words could be utilised to inform the conversational negation of evolving meanings. The conversational negation of multiple words allows us to calculate the similarity between actors. One aspect that this example does not consider is interacting meanings; DisCoCirc allows multiple meanings to interact via updates, such as the transitive verb `love` which connects two wires.

6.3.1.3 Interacting wires

This general approach is still applicable to interacting wires. Let us consider the three sentences:

`Alice is alone.`

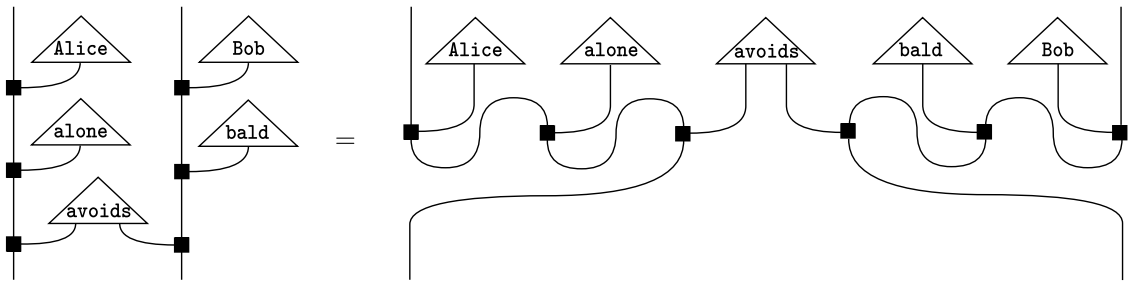
`Bob is bald.`

`Alice avoids Bob.`

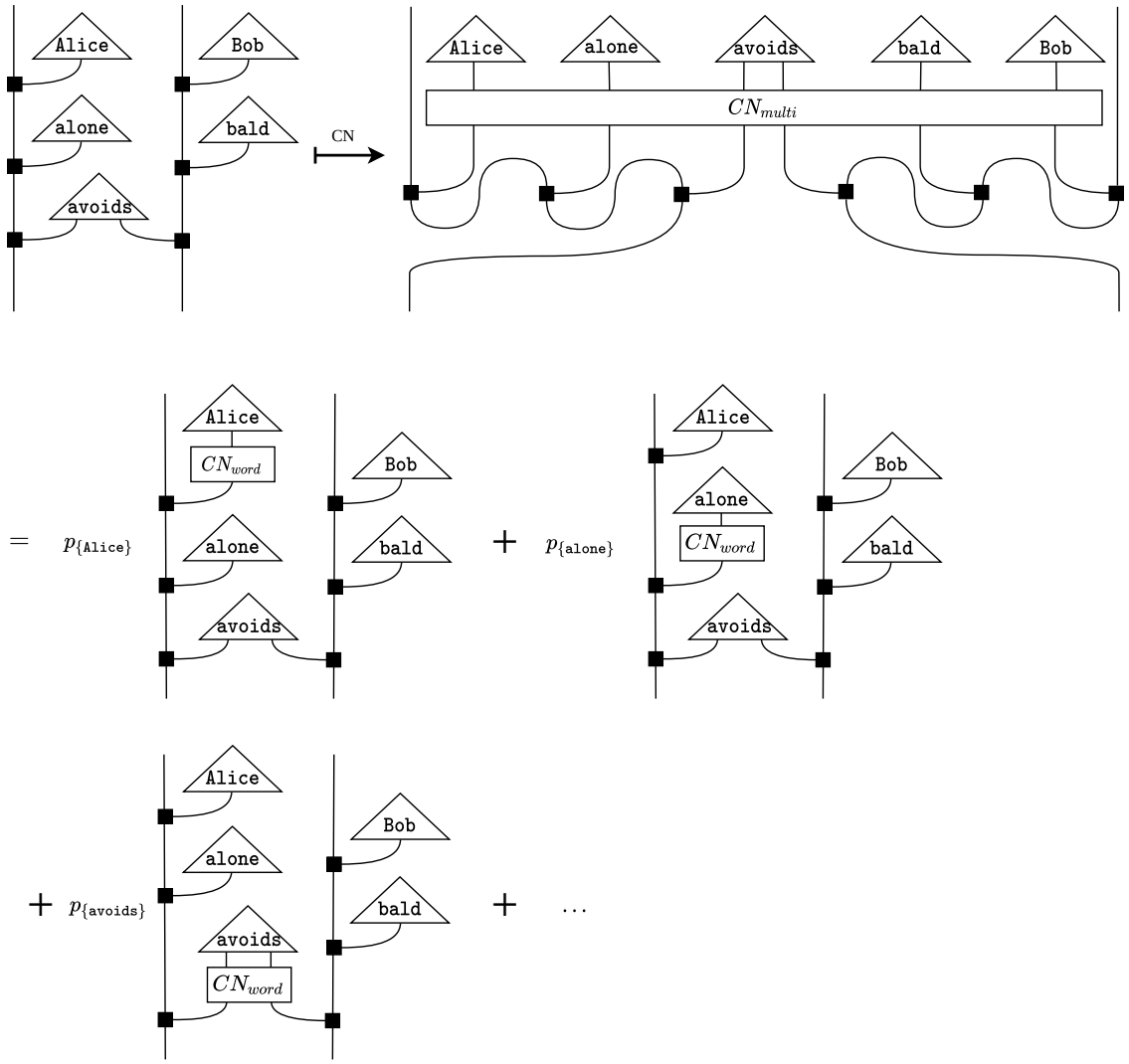
These sentences have the same diagrammatic representation as the sentence:

`Alice, who is alone, avoids Bob, who is bald.`

Once more we can use the yanking equations to get the two equal circuits:



Thus someone similar to "not `Alice`" might be someone who is `with people` but avoids `bald Bob` (negation set {`alone`}). It might alternatively be someone who is `alone` but avoids `bald Dave` (negation set {`Dave`}). Similar to the previous case, someone who is similar to `Alice` shares some attributes with her and has other attributes that are different. Thus the negation of `Alice` can be computed analogously. We have:



In this particular story, `Alice` and `Bob` are entangled with each other. Therefore their respective negations have identical diagrammatic representations, namely the one above. For both `Alice` and `Bob` their negation corresponds to negating a subset of the updates to `Alice` and `Bob`. However, they are not identical, as the corresponding weights for the negation sets differ. The more plausible negation sets for `Bob` are not the same as the more plausible negation sets for `Alice`.

We have to observe that with interacting wires, the previous example of entailment measures becomes more complicated. The dimension of an actor depends on how many different interactions they have. Thus the representation of two actors and their relations might have different dimensions. Therefore current entailment measures are not applicable. This challenge is left for future work.

Utilising the negation of multiple words for the negation of evolving meaning is helpful to inform the plausibilities. However, it cannot, by itself, be utilised as a negation in the DisCoCirc framework, as it removes the positive instance of the evolving meaning being negated. In our example, the wire which used to contain `Alice` now contains something similar but not equal to `Alice`. In fact, when `Alice` interacts with other evolving meanings, such as `Bob` in our second example, negating `Alice` also removes the positive instance of `Bob`. Therefore, after the negation, a simple sentence such as "`Alice is happy.`" could not be modelled as `Alice` has been replaced by "not `Alice`". Thus we propose to solely utilise this method to inform the weights of the conversational negation framework propose earlier in this section. The actual negation of evolving meanings is modelled via a change in the update structure of the sentence.

Putting it together

7

Conversational negation of sentences

Contents

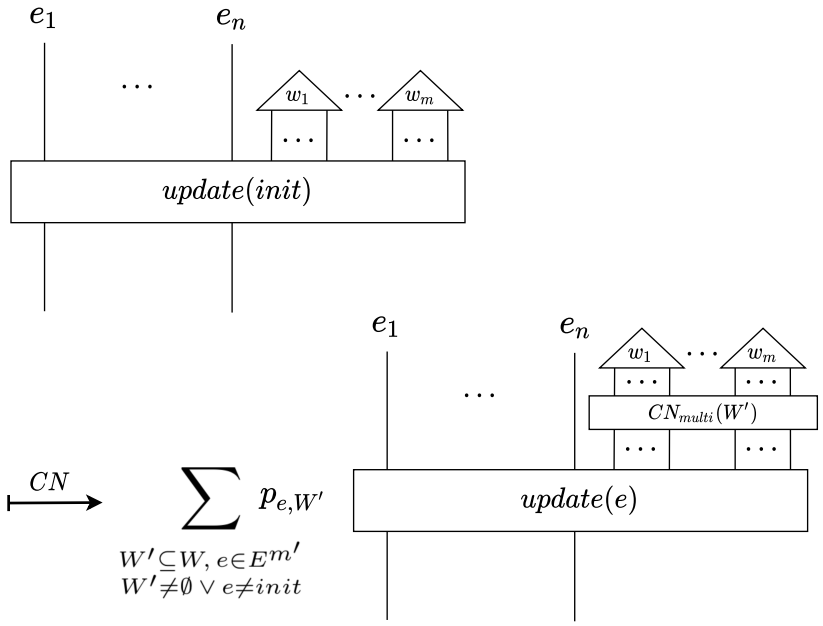
7.1 Intuition

A sentence in DisCoCirc acts on evolving meanings, which are being updated by some meaning states. We have now proposed three different conversational negations respectively for words, multiple words and evolving meanings. While the negations of words and multiple words change the meaning states, they do not affect the update structure. On the other hand, the negation of evolving meanings only changes the update structure, leaving the meaning states unaffected. Therefore, to fully model the negation of a sentence in the DisCoCirc framework, we have to combine these negations to form a new framework that changes both the meaning states and the update structure.

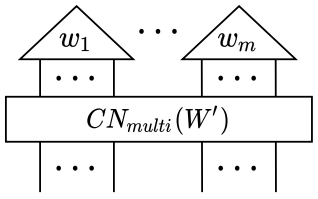
The negation of a sentence must sum over all possible interpretations of the sentence, which are not identical to the original, positive statement. Being different from the original can either mean being different with regards to the meaning states, different with regards to the updates or different with regards to both.

7.2 The framework

Let us consider a sentence of m words, with m' updates acting in a story of n actors with an initial update structure $init \in E^{m'}$. We negate this sentence by applying the CN function, which maps it as follows:



where the operation $update(e)$ takes the meaning states and updates the corresponding evolving meanings as defined by the parameter e. The operation $CN_{multi}(W')$ corresponds to a single summand of the CN_{multi} negation, namely the one under the negation set W'. More formally we have:

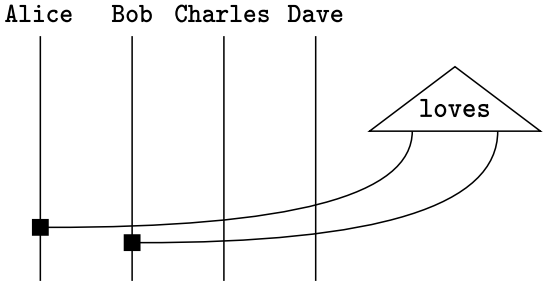


$$:= \quad \bigotimes_{i=1}^{m} \begin{cases} w_i & \text{if } w_i \notin W' \\ CN_{word}(w_i) & \text{if } w_i \in W' \end{cases}$$

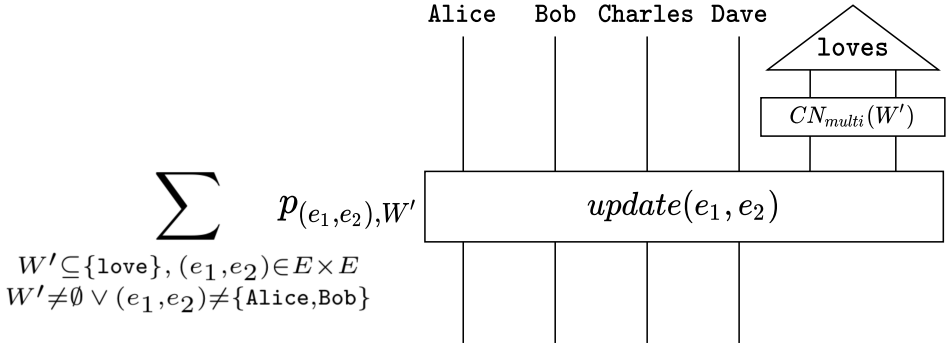
We observe that we either negate a meaning state in its entirety or not at all. Thus, for example, for the transitive verb `love` we either negate both outputs or neither. It is not possible to negate only one output of a given meaning state.

For $e = init \wedge W' \neq \emptyset$ the framework is identical to CN_{multi}, i.e. we only change the meaning states. For $e \neq init \wedge W' = \emptyset$, the negation is identical the negation of evolving meanings. When $e \neq init \wedge W' \neq \emptyset$, we combine both frameworks.

As an example, we will negate the sentence "`Alice loves Bob`" where `Alice` and `Bob` are evolving meanings in a text that additionally talks about `Charles` and `Dave`. The positive sentence thus looks like:

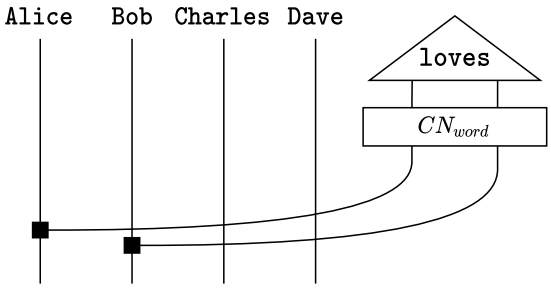


After applying the negation function, we get:



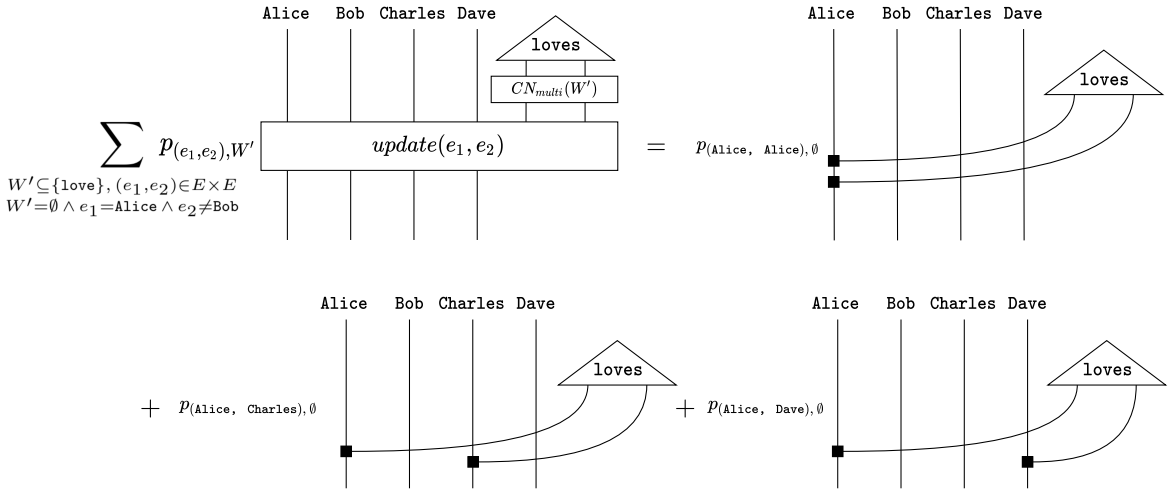
where $E = \{\texttt{Alice, Bob, Charles, Dave}\}$.

The interpretation of the sentence as "`Alice does not love Bob`", i.e. the negation set $\{\texttt{love}\}$, would then correspond to $W' = \{\texttt{love}\}, (e_1, e_2) = init = (\texttt{Alice}, \texttt{Bob})$. It thus corresponds to only negating the static meaning of `love` and leaving all other aspects as is. Diagrammatically we have:



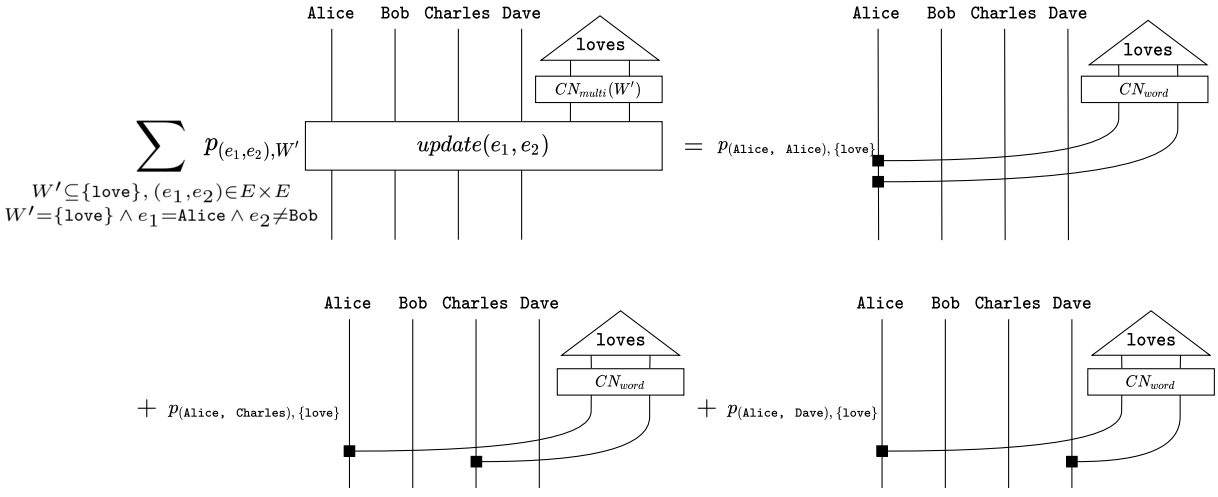
We observe that $CN_{multi}(\{\texttt{love}\})$ resolves to applying the conversational negation of words to the meaning state `love`.

The negation set $\{\texttt{Bob}\}$ corresponds to $W' = \emptyset, e_1 = \texttt{Alice}, e_2 \neq \texttt{Bob}$, i.e. leaving the meaning states as is and iterating over all update structures where the first target is `Alice` and the second target is not `Bob`. We have:



$CN_{multi}(\emptyset)$ resolves to the identity, i.e. not doing anything. Thus this interpretation corresponds to the negation set $\{\texttt{Bob}\}$ in Chapter 6.

Larger negation sets such as $\{\texttt{love}, \texttt{Bob}\}$ correspond to $W' = \{\texttt{love}\}, e_1 = \texttt{Alice}, e_2 \neq \texttt{Bob}$, i.e. both negating the static meaning of `love` and iterating over all update structures where the first update is to `Alice` and the second update is to someone other than `Bob`. We have:



All these different interpretations are graded via the weights $p_{e,W'}$, which can be derived from various sources of context.

Where to go next?

8 Conclusion

Contents

8.1 Overview

In this thesis, we have modelled the *search-for-alternatives-view* on conversational negation of different scopes. Each of our negation frameworks are built on the underlying hypothesis that:

conversational negation is context dependent

This hypothesis manifests itself in our frameworks through weighted sums. We use ambiguity, captured by weighted sums, to create a negation for all potential contexts.

Table 8.1 gives an overview of the four different scopes of negation we have modelled and how they interact with the context. For the negation of words, we sum directly over different contexts that inform the alternatives elicited by the negation. For the negation of multiple words, the context is not explicitly utilised in the framework. Instead, it is used to inform the weights of the sum.

Table 8.1: Overview over the proposed negation frameworks of various scopes

Scope	Summands	Role of context	Affects
Words	Contexts	Directly incorporated	Meaning states
Multiple words	Target of negation	Informs weights	Meaning states
Evolving meanings	Alternative evolving meanings	Informs weights	Update structure
Sentences	Interpretation of negation	Informs weights	Both

Each summand corresponds to different targets of the negation, identified by the respective negation sets. Similarly, for the negation of evolving meanings, the context is not an explicit part of the operation anymore. Instead, we rely on external calculations to extract the relevant information from the context, in this case, the intended alternative to an evolving meaning. Our final framework for conversational negation combines all our proposals into one operation. The core observation is that we have negations acting on the meaning states and negations acting on the update structure. Combining them, we propose an operation that can be applied to any sentence modelled in the DisCoCirc framework.

We observe that only for the negation of words, we propose to incorporate the context as part of the framework directly. For the other negations, we utilise our understanding of the negation to create the possible interpretations over which we sum. The context is then used to inform the plausibility of each interpretation which is captured in the weights.

While this thesis and the two accompanying papers — Rodatz et al. (2021) and Shaikh et al. (2021) — make good strides towards conversational negation, they raise new questions for future work.

We observed that negations are dependent on context, which is not necessarily explicit in a given text. Therefore, we proposed using external sources of context to derive and grade the different interpretations of a negation. We introduce this mechanism to the DisCoCirc framework for the purpose of negation. For

all negation frameworks, we would like to explore additional sources of context. One overarching challenge is to find meaning representations that can embed some of these sources of context directly, therefore reducing the reliance on the external derivation of the context. For example, this could be explored utilising conceptual spaces (Bolt et al., 2019).

8.2 Negation of words

In the case of the negation of words, we propose four different frameworks. Additionally, we propose a method to create and weigh the contexts. We validate the frameworks and the context creations experimentally. These results give us some insights into which approaches do not work, such as taking the logical negation or certain composition operations in the negation frameworks CN_{word1} and CN_{word2}. However, we need to create further experiments to compare frameworks that perform well at our experiments. Further experiments should aim to differentiate the high-performing frameworks based on additional properties expected from a negation. Additionally, we should explore alternative implementations. Especially the context creation for the negation of words could profit from additional work. Finally, methods to implement the proposal in Section 4.3.2 to remove the necessity of external entailment hierarchies should be created and validated.

We would also like to explore the disambiguation of negations throughout a text. Ambiguity in a negation takes two parts; (1) negating ambiguous meanings and (2) the ambiguity introduced by the negation. The former is already partially incorporated into the framework by considering different interpretations of a word in the context. However, the ambiguity introduced by the negation through summing over multiple contexts has received less attention. Ideally, the remaining meaning updates from the text serve to disambiguate the right choice of context, similar to how meaning update disambiguate meanings. While the current experimental setup allowed for some small scale experiments to confirm this intuition, larger, more structured experiments have to be conducted to explore the disambiguation of negations throughout a text.

8.3 Negation of multiple words

For the conversational negation of multiple words, we propose a framework utilising the negation of individual words. We additionally propose a method to derive the weights from the surrounding text. Both the negation framework and the method to derive the weights should be experimentally validated. Being a comparatively new proposal, the DisCoCirc framework, in contrast to the older DisCoCat framework, has little experimental validation. As a result, there are still implementation details concerning the broader framework to be solved. These details have to be solved before the negation of multiple words can be explored experimentally.

Additionally, we should explore other methods of extracting the weights for the negation. These include, among others, grammatical structure, the location of the speaker and the intention of the speaker.

Our model only focuses on the negation of single word constituents. Therefore the following alternative to the sentence cannot be modelled:

`Bob does not publish a paper - Bob is lazy`

where the negated constituent is `publish a paper`, which elicits the alternative of being `lazy`. We focused on single word constituents for the purpose of clarity and to be able to utilise the previously defined negation of words. However, the framework can be extended to contain multi-word constituents by adding additional summands. The two main challenges to overcome are (1) finding the constituents and (2) calculating the negation of a multi-word constituent. For the first challenge, constituent trees could be a helpful tool (Anderson, 2018, Chapter 8). For the second challenge, the negation of words framework has to be extended to handle multi-word constituents. Expanding the framework to multi-word constituents could also enable us to model the negation of non-conjunctive composition as exemplified by the "pet fish" problem (Coecke & Lewis, 2015).

8.4 Negation of evolving meanings

For the conversational negation of evolving meanings, we propose a framework that changes the update structure. Additionally, we propose a method to derive the weights based on the negation of multiple words. Both the framework as well as the context creation are not yet experimentally validated. They, too, rely on the DisCoCirc framework.

Another challenge we have to solve to negate evolving meanings is incorporating more complex update structures, which will probably be necessary for some grammatical constructions. Here we rely on a formalisation of the update structures, which is currently being explored. The first steps towards the final product are, for example, made in Coecke and Wang (2021).

8.5 Negation of sentences

The negation of sentences combines the negation of multiple words with the negation of evolving meanings to change the meaning states and the update structure. While we have proposed methods to derive weights throughout the different negation frameworks, many still have to be formalised and experimentally validated.

Similar to the negation of multiple words, we would like to explore multi-constituent negation. In particular, negating an evolving meaning and a meaning state simultaneously. For example, we could have the following interpretation:

`Alice is not playing with Bob - She is sleeping`

This interpretation simultaneously negates the meaning state for `playing` and the evolving meaning of `Bob`. In this case, it removes the update to `Bob`, while negating the meaning state.

8.6 Final remarks

Overall we have proposed a series of frameworks for conversational negation with growing scope — each relying on intuitions gained from the previous. The main challenges lie in formalising the derivation of weights and experimental validation.

Considering future work, we would also like to focus on other terms of conversational logic, such as `and`, `or` and `all`. For this, the action of these logical elements has to be explored, and their interaction with each other and negation formalised. A long-term goal is to propose a broader framework for *conversational logic* for compositional, distributional semantics.

References

Abramsky, S., & Coecke, B. (2004). A categorical semantics of quantum protocols. *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science, 2004.*, 415–425. https://doi.org/10.1109/LICS.2004.1319636

Ahsaee, M. G., Naghibzadeh, M., & Naeini, S. E. Y. (2014). Semantic similarity assessment of words using weighted wordnet. *International Journal of Machine Learning and Cybernetics*, *5*(3), 479–490. https://www.researchgate.net/publication/326304723_WordNet-based_Semantic_Similarity_Measures_for_Process_Model_Matching

Anderson, C. (2018). *Essentials of linguistics*. McMaster University. https://ecampusontario.pressbooks.pub/essentialsoflinguistics/front-matter/introduction/

Ashoush, D. (2015). *Categorical models of meaning: Accommodating for lexical ambiguity and entailment* (Master's thesis). University of Oxford. http://www.cs.ox.ac.uk/people/bob.coecke/Daniela.pdf

Baksalary, J. K., Pukelsheim, F., & Styan, G. P. (1989). Some properties of matrix partial orderings. *Linear Algebra and its Applications*, *119*, 57–85. https://doi.org/https://doi.org/10.1016/0024-3795(89)90069-4

Balkir, E., Sadrzadeh, M., & Coecke, B. (2016). Distributional sentence entailment using density matrices. In M. T. Hajiaghayi & M. R. Mousavi (Eds.), *Topics in theoretical computer science* (pp. 1–22). Springer International Publishing. https://doi.org/10.1007/978-3-319-28678-5_1

Bankova, D., Coecke, B., Lewis, M., & Marsden, D. (2019). Graded hyponymy for compositional distributional semantics. *Journal of Language Modelling*, *6*(2), 225–260. https://doi.org/10.15398/jlm.v6i2.230

Baroni, M. (2013). Composition in distributional semantics. *Language and Linguistics Compass*, *7*(10), 511–522. https://doi.org/https://doi.org/10.1111/lnc3.12050

Bellegarda, J. R. (2000). Exploiting latent semantic information in statistical language modeling. *Proceedings of the IEEE*, *88*(8), 1279–1296. https://doi.org/10.1109/5.880084

Bolt, J., Coecke, B., Genovese, F., Lewis, M., Marsden, D., & Piedeleu, R. (2019). Interacting conceptual spaces i: Grammatical composition of concepts. In M. Kaipainen, F. Zenker, A. Hautamäki, & P. Gärdenfors (Eds.), *Conceptual spaces: Elaborations and applications* (pp. 151–181). Springer International Publishing. https://doi.org/10.1007/978-3-030-12800-5_9

Boyd-Graber, J., Fellbaum, C., Osherson, D., & Schapire, R. (2005). Adding dense, weighted connections to wordnet [3rd International Global WordNet Conference, GWC 2006 ; Conference date: 22-01-2006 Through 26-01-2006]. *GWC 2006*, 29–35.

Coecke, B., & Paquette, É. (2011). Categories for the practising physicist. In B. Coecke (Ed.), *New structures for physics* (pp. 173–286). Springer. https://doi.org/10.1007/978-3-642-12821-9_3

Coecke, B. (2020). The mathematics of text structure. https://arxiv.org/abs/1904.03478

Coecke, B., de Felice, G., Meichanetzidis, K., & Toumi, A. (2020). Foundations for near-term quantum natural language processing. *arXiv preprint arXiv:2012.03755*. https://arxiv.org/abs/2012.03755

Coecke, B., & Kissinger, A. (2017). *Picturing quantum processes: A first course in quantum theory and diagrammatic reasoning*. Cambridge University Press. https://doi.org/10.1017/9781316219317

Coecke, B., & Lewis, M. (2015). A compositional explanation of the 'pet fish'phenomenon. *International Symposium on Quantum Interaction*, 179–192.

Coecke, B., & Meichanetzidis, K. (2020). Meaning updating of density matrices. *FLAP*, *7*, 745–770. https://arxiv.org/abs/2001.00862

Coecke, B., Sadrzadeh, M., & Clark, S. (2010). Mathematical foundations for a compositional distributional model of meaning. In J. van Benthem, M. Moortgat, & W. Buszkowski (Eds.), *A festschrift for jim lambek* (pp. 345–384). https://arxiv.org/abs/1003.4394

Coecke, B., & Wang, V. (2021). Grammar equations. *arXiv preprint arXiv:2106.07485*. https://arxiv.org/abs/2106.07485

De las Cuevas, G., Klinger, A., Lewis, M., & Netzer, T. (2020). Cats climb entails mammals move: Preserving hyponymy in compositional distributional semantics. *Proceedings of SEMSPACE 2020*. https://arxiv.org/abs/2005.14134

Duneau, T. (2021). Parsing conjunctions in discocirc. *Proceedings of the 2021 Workshop on Semantic Spaces at the Intersection of NLP, Physics, and Cognitive Science (SemSpace)*, 66–75. https://iwcs2021.github.io/proceedings/semspace/index.html

Evans, J. S. B. T. (1989). *Bias in human reasoning: Causes and consequences.* Lawrence Erlbaum Associates, Inc.

Evans, J. S. B. T. (1996). Deciding before you think: Relevance and reasoning in the selection task. *British Journal of Psychology*, *87*(2), 223–240. https://doi.org/https://doi.org/10.1111/j.2044-8295.1996.tb02587.x

Firth, J. R. (1957). A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*. Reprinted in: Palmer, F. R. (ed.) (1968). Selected Papers of J. R. Firth 1952-59, pages 168-205. Longmans, London.

Foltz, P. W., Kintsch, W., & Landauer, T. K. (1998). The measurement of textual coherence with latent semantic analysis. *Discourse processes*, *25*(2-3), 285–307. https://doi.org/10.1080/01638539809545029

Grefenstette, E., & Sadrzadeh, M. (2011). Experimental support for a categorical compositional distributional model of meaning. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 1394–1404. https://aclanthology.org/D11-1129

Hansen, D. L., Shneiderman, B., Smith, M. A., & Himelboim, I. (2020). Chapter 4 - installation, orientation, and layout. In D. L. Hansen, B. Shneiderman, M. A. Smith, & I. Himelboim (Eds.), *Analyzing social media networks with nodexl (second edition)* (Second Edition, pp. 55–66). Morgan Kaufmann. https://doi.org/https://doi.org/10.1016/B978-0-12-817756-3.00004-2

Hearst, M. A. (1992). Automatic acquisition of hyponyms from large text corpora. *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, 539–545. https://doi.org/10.3115/992133.992154

Hermann, K. M., Grefenstette, E., & Blunsom, P. (2013). "not not bad" is not "bad": A distributional account of negation. *Proceedings of the 2013 Workshop on*

Continuous Vector Space Models and their Compositionality. https://arxiv.org/abs/1306.2158

Heunen, C., & Vicary, J. (2018). Categorical quantum mechanics: An introduction. Retrieved on 2021/07/25 from https://www.cs.ox.ac.uk/files/10510/notes.pdf.

Horn, L. (1972). On the semantic properties of logical operators in english. *Unpublished Ph.D. dissertation.* https://www.researchgate.net/publication/247046187_On_the_Semantic_Properties_of_Logical_Operators_in_English

Kartsaklis, D., & Sadrzadeh, M. (2013). Prior disambiguation of word tensors for constructing sentence vectors. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1590–1601. Retrieved on 2021/08/21 from https://www.cs.ox.ac.uk/files/5725/karts_sadr_emnlp.pdf.

Kartsaklis, D., & Sadrzadeh, M. (2014). A study of entanglement in a categorical framework of natural language. *Electronic Proceedings in Theoretical Computer Science*, *172*, 249–261. https://doi.org/10.4204/eptcs.172.17

Kartsaklis, D., & Sadrzadeh, M. (2016a). A compositional distributional inclusion hypothesis. In M. Amblard, P. de Groote, S. Pogodalla, & C. Retoré (Eds.), *Logical aspects of computational linguistics. celebrating 20 years of lacl (1996–2016)* (pp. 116–133). Springer. https://doi.org/10.1007/978-3-662-53826-5_8

Kartsaklis, D., & Sadrzadeh, M. (2016b). Distributional inclusion hypothesis for tensor-based composition. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2849–2860. https://aclanthology.org/C16-1268

Kartsaklis, D., Sadrzadeh, M., Pulman, S., & Coecke, B. (2013). Reasoning about meaning in natural language with compact closed categories and frobenius algebras. In J. Chubb, A. Eskandarian, & V. Harizanov (Eds.), *Logic and algebraic structures in quantum computing and information.* Cambridge University Press. Retrieved on 2021/08/21 from http://www.cs.ox.ac.uk/publications/publication6598-abstract.html.

Kartsaklis, D., Sadrzadeh, M., Pulman, S., & Coecke, B. (2016). Reasoning about meaning in natural language with compact closed categories and frobenius algebras. In J. Chubb, A. Eskandarian, & V. Harizanov (Eds.), *Logic and algebraic structures in quantum computing* (pp. 199–222). Cambridge University Press. https://doi.org/10.1017/CBO9781139519687.011

Kornai, A. (2011). Probabilistic grammars and languages. *Journal of Logic, Language and Information*, *20*, 317–328. https://doi.org/10.1007/s10849-011-9135-z

Kruszewski, G., Paperno, D., Bernardi, R., & Baroni, M. (2016). There is no logical negation here, but there are alternatives: Modeling conversational negation with distributional semantics. *Computational Linguistics*, *42*(4), 637–660. https://doi.org/10.1162/COLI_a_00262

Lambek, J. (1999). Type grammar revisited. In A. Lecomte, F. Lamarche, & G. Perrier (Eds.), *Logical aspects of computational linguistics* (pp. 1–27). Springer.

Landauer, T. K., Laham, D., Rehder, B., & Schreiner, M. E. (1997). How well can passage meaning be derived without using word order? a comparison of latent semantic analysis and humans. *Proceedings of the 19th annual meeting of the Cognitive Science Society*, 412–417.

Lee, E. N. (1972). Plato on negation and not-being in the sophist. *The Philosophical Review*, *81*(3), 267–304. http://www.jstor.org/stable/2184327

Lewis, M. (2019). Compositional hyponymy with positive operators. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, 638–647. https://doi.org/10.26615/978-954-452-056-4_075

Lewis, M. (2020). Towards logical negation for compositional distributional semantics. *IfCoLoG Journal of Logics and their Applications*, *7*(3). https://arxiv.org/abs/2005.04929

Lorenz, R., Pearson, A., Meichanetzidis, K., Kartsaklis, D., & Coecke, B. (2021). Qnlp in practice: Running compositional models of meaning on a quantum computer. https://arxiv.org/abs/2102.12846

McCarthy, D., Koeling, R., Weeds, J., & Carroll, J. (2004). Finding predominant word senses in untagged text. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, 279–286. https://doi.org/10.3115/1218955.1218991

Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*, *38*(11), 39–41. https://doi.org/10.1145/219717.219748

Mitchell, J., & Lapata, M. (2010). Composition in distributional models of semantics. *Cognitive Science*, *34*(8), 1388–1429. https://doi.org/https://doi.org/10.1111/j.1551-6709.2010.01106.x

Mohammad, S. M., Dorr, B. J., Hirst, G., & Turney, P. D. (2013). Computing Lexical Contrast. *Computational Linguistics*, *39*(3), 555–590. https://doi.org/10.1162/COLI_a_00143

Oaksford, M., & Stenning, K. (1992). Reasoning with conditionals containing negated constituents. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *18*(4), 835–854. https://doi.org/10.1037/0278-7393.18.4.835

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. http://www.aclweb.org/anthology/D14-1162

Piedeleu, R., Kartsaklis, D., Coecke, B., & Sadrzadeh, M. (2015). Open system categorical quantum semantics in natural language processing. *arXiv preprint arXiv:1502.00831*. http://arxiv.org/abs/1502.00831

Prado, J., & Noveck, I. A. (2006). How reaction time measures elucidate the matching bias and the way negations are processed. *Thinking & Reasoning*, *12*(3), 309–328. https://doi.org/10.1080/13546780500371241

Rodatz, B., Shaikh, R., & Yeh, L. (2021). Conversational negation using worldly context in compositional distributional semantics. *Proceedings of the 2021 Workshop on Semantic Spaces at the Intersection of NLP, Physics, and Cognitive Science (SemSpace)*, 53–65. https://arxiv.org/abs/2105.05748

Sadrzadeh, M., Clark, S., & Coecke, B. (2013). The Frobenius anatomy of word meanings I: subject and object relative pronouns. *Journal of Logic and Computation*, *23*(6), 1293–1317. https://doi.org/10.1093/logcom/ext044

Sadrzadeh, M., Clark, S., & Coecke, B. (2014). The Frobenius anatomy of word meanings II: possessive relative pronouns *. *Journal of Logic and Computation*, *26*(2), 785–815. https://doi.org/10.1093/logcom/exu027

Sadrzadeh, M., Kartsaklis, D., & Balkır, E. (2018). Sentence entailment in compositional distributional semantics. *Annals of Mathematics and Artificial Intelligence*, *82*(4), 189–218. https://doi.org/10.1007/s10472-017-9570-x

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Commun. ACM*, *18*(11), 613–620. https://doi.org/10.1145/361219.361220

Schütze, H. (1998). Automatic word sense discrimination. *Comput. Linguist.*, *24*(1), 97–123. https://dl.acm.org/doi/10.5555/972719.972724

Selinger, P. (2011). A survey of graphical languages for monoidal categories. In B. Coecke (Ed.), *New structures for physics* (pp. 289–355). Springer. https://doi.org/10.1007/978-3-642-12821-9_4

Selinger, P. (2007). Dagger compact closed categories and completely positive maps: (extended abstract) [Proceedings of the 3rd International Workshop on Quantum Programming Languages (QPL 2005)]. *Electronic Notes in Theoretical Computer Science*, *170*, 139–163. https://doi.org/10.1016/j.entcs.2006.12.018

Shaikh, R. A., Yeh, L., Rodatz, B., & Coecke, B. (2021). Composing conversational negation [To appear in Proceedings of ACT 2021]. https://arxiv.org/abs/2107.06820

Tull, S. (2021). A categorical semantics of fuzzy concepts in conceptual spaces. Retrieved on 2021/08/21 from https://www.cl.cam.ac.uk/events/act2021/papers/ACT_2021_paper_66.pdf.

Van de Wetering, J. (2016). Entailment relations on distributions. *Electronic Proceedings in Theoretical Computer Science*, *221*, 58–66. https://doi.org/10.4204/eptcs.221.7

Van de Wetering, J. (2018). Ordering quantum states and channels based on positive bayesian evidence. *Journal of Mathematical Physics*, *59*(10), 102201. https://doi.org/10.1063/1.5023474

Widdows, D., & Peters, S. (2003). Word vectors and quantum logic: Experiments with negation and disjunction. *Mathematics of Language*, *8*, 141–154. https://www.puttypeg.net/papers/quantum-senses.pdf

Yeung, R., & Kartsaklis, D. (2021). A ccg-based version of the discocat framework. *Proceedings of the 2021 Workshop on Semantic Spaces at the Intersection of NLP, Physics, and Cognitive Science (SemSpace)*, 20–31. https://iwcs2021.github.io/proceedings/semspace/index.html

Bibliography

The provided citation guide[1] requires us to differentiate between references and bibliography. As all relevant readings for this thesis were incorporated into the references, my bibliography is identical to my references. Therefore it is omitted.

[1] http://www.cs.ox.ac.uk/files/3161/Referencing.pdf

Appendices

A Proofs

A.1 Negation Properties

A.1.1 Double negative

A.1.1.1 The subtraction-from-identity-negation

Theorem 3. *The subtraction-from-identity-negation obeys the double negative. For any positive operator* A *we have:*

$$\neg_{sub}(\neg_{sub}\mathsf{A}) = \mathsf{A} \tag{A.1}$$

Proof. We have:

$$\begin{aligned} \neg_{sub}(\neg_{sub}A) &= \neg_{sub}(\mathbb{I} - \mathsf{A}) && \text{(A.2)} \\ &= \mathbb{I} - (\mathbb{I} - \mathsf{A}) && \text{(A.3)} \\ &= (\mathbb{I} - \mathbb{I}) + \mathsf{A} && \text{(A.4)} \\ &= A && \text{(A.5)} \end{aligned}$$

□

A.1.1.2 The support-inverse-negation

Theorem 4. *The support-inverse-negation obeys the double negative. For any positive operator* A *we have:*

$$\neg_{supp}(\neg_{supp}\mathsf{A}) = \mathsf{A} \tag{A.6}$$

Proof. For any positive operators A with spectral decomposition $\mathsf{A} = \sum_i \lambda_i |i\rangle \langle i|$. We have $\mathsf{A}' \coloneqq \neg_{supp}\mathsf{A}$ has the same eigenbasis. Thus A' has spectral decomposition $\sum_i \lambda'_i |i\rangle \langle i|$ for some set of eigenvalues λ'. Therefore $\neg_{supp}\mathsf{A}' = \sum_i \lambda''_i |i\rangle \langle i| =: \mathsf{A}''$ once more has the same eigenbasis. Thus, to show that $\mathsf{A} = \mathsf{A}''$ we have to show that for all i we have $\lambda_i = \lambda''_i$.

For $\lambda_i = 0$, we have $\lambda'_i = 0$ by the definition of $\neg_{supp}$ and thus similarly $\lambda''_i = 0 = \lambda_i$.

For $\lambda_i > 0$, we have $\lambda'_i = \frac{1}{\lambda_i} > 0$ and thus $\lambda''_i = \frac{1}{\lambda'_i} = \lambda_i$.

Thus in both cases $\lambda_i = \lambda''_i$ and therefore we have shown that $\mathsf{A} = \mathsf{A}''$. □

A.1.1.3 The kernel-inverse-negation

The *kernel-inverse-negation* does not obey the double negative.

We give a simple counter example. Let:

$$\mathsf{A} = \begin{pmatrix} 0.5 & 0 \\ 0 & 0 \end{pmatrix} \tag{A.7}$$

This matrix has eigenvectors:

$$X_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad X_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{A.8}$$

with respective eigenvalues $\lambda_1 = 0.5$ and $\lambda_2 = 0$. Thus we have:

$$\neg_{ker}\mathsf{A} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{A.9}$$

with eigenvalues $\lambda'_1 = 0$ and $\lambda'_2 = 1$ for the same eigenvectors as A. But then:

$$\neg_{ker}(\neg_{ker}\mathsf{A}) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \neq \mathsf{A} \tag{A.10}$$

Thus $\neg_{ker}$ does not obey the double negative. Applying the *kernel-inverse-negation* twice results in the identity over the support of the original matrix.

A.1.1.4 The inverse-negation

The *inverse-negation* does not obey the double negative.

We give a simple counter example. Let:

$$\mathsf{A} = \begin{pmatrix} 0.1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{A.11}$$

This matrix has eigenvectors:

$$X_1 = \begin{pmatrix}1\\0\\0\end{pmatrix}, \quad X_2 = \begin{pmatrix}0\\1\\0\end{pmatrix}, \quad X_3 = \begin{pmatrix}0\\0\\1\end{pmatrix} \tag{A.12}$$

with respective eigenvalues $\lambda_1 = 0.1, \lambda_2 = 1$ and $\lambda_3 = 0$. We have:

$$\neg_{supp}\mathsf{A} = \begin{pmatrix}10 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0\end{pmatrix} \tag{A.13}$$

which we normalise to get:

$$normalise(\neg_{supp}\mathsf{A}) = \begin{pmatrix}1 & 0 & 0\\0 & 0.1 & 0\\0 & 0 & 0\end{pmatrix} \tag{A.14}$$

Additionally we have:

$$\neg_{ker}\mathsf{A} = \begin{pmatrix}0 & 0 & 0\\0 & 0 & 0\\0 & 0 & 1\end{pmatrix} \tag{A.15}$$

Thus:

$$\neg_{inv}\mathsf{A} = normalise(\neg_{supp}\mathsf{A}) + \neg_{ker}\mathsf{A} = \begin{pmatrix}1 & 0 & 0\\0 & 0.1 & 0\\0 & 0 & 1\end{pmatrix} \tag{A.16}$$

But then, as the kernel of $\neg_{inv}A$ is empty, we have:

$$\neg_{inv}(\neg_{inv}\mathsf{A}) = normalise(\neg_{supp}(\neg_{inv}A)) = \begin{pmatrix}0.1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 0.1\end{pmatrix} \neq \mathsf{A} \tag{A.17}$$

Thus $\neg_{inv}$ does not obey the double negative.

A.1.2 Contrapositive

During the thesis, we abuse the notation of entailment by not always clearly specifying the concrete entailment measure we use. This was done as the choice of entailment is always sufficiently clear from the context. Throughout these proofs, we use a slightly adapted notation, where we specify the choice of entailment. For two positive operators A, B, we will write $A \sqsubseteq_k^{k_{\mathsf{hyp}}} B$ to symbolise an entailment of strength k from A to B under the entailment measure k_{hyp}. A similar notation will be used for k_{E} and k_{BA}.

A.1.2.1 The subtraction-from-identity-negation

$k_{\textbf{hyp}}$ - ✓

Theorem 5. *For two positive operators A and B we have:*

$$A \sqsubseteq_k^{k_{hyp}} \quad \Longleftrightarrow \quad \neg_{sub} B \sqsubseteq_k^{k_{hyp}} \neg_{sub} A \tag{A.18}$$

when $k = 1$

Proof. We have:

$$\mathsf{A} \sqsubseteq_k^{k_{\mathsf{hyp}}} \mathsf{B} \quad \Longleftrightarrow \quad \mathsf{B} - k\mathsf{A} \geq 0 \tag{A.19}$$

where we use "≥ 0" to denote that an operator is positive.

Thus:

$$\neg_{sub}\mathsf{B} \sqsubseteq_k^{k_{\mathsf{hyp}}} \neg_{sub}\mathsf{A} = (\mathbb{I} - \mathsf{B}) \sqsubseteq_k^{k_{\mathsf{hyp}}} (\mathbb{I} - \mathsf{A}) \tag{A.20}$$

$$\Longleftrightarrow (\mathbb{I} - \mathsf{A}) - k \cdot (\mathbb{I} - \mathsf{B}) \geq 0 \tag{A.21}$$

$$\Longleftrightarrow (1 - k) \cdot \mathbb{I} - \mathsf{A} + k \cdot \mathsf{B} \geq 0 \tag{A.22}$$

But for $k = 1$ we have:

$$(1 - k) \cdot \mathbb{I} - \mathsf{A} + k \cdot \mathsf{B} = \mathsf{B} - \mathsf{A} \tag{A.23}$$

Thus, for $k = 1$, we have:

$$(\mathbb{I} - \mathsf{A}) - k \cdot (\mathbb{I} - \mathsf{B}) \geq 0 \quad \Longleftrightarrow \quad \mathsf{B} - \mathsf{A} \geq 0 \quad \Longleftrightarrow \quad \mathsf{A} \sqsubseteq_1^{k_{\mathsf{hyp}}} \mathsf{B} \tag{A.24}$$

and therefore:

$$\mathsf{A} \sqsubseteq_1^{k_{\mathsf{hyp}}} \mathsf{B} \quad \Longleftrightarrow \quad \neg_{sub}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{hyp}}} \neg_{sub}\mathsf{A} \tag{A.25}$$

□

k_{E} - ✓ Lewis (2019) shows that the *subtraction-from-identity-negation* does not preserves the contrapositive for k_{E} in general.

However, we have:

Theorem 6. *For two positive operators A and B we have:*

$$A \sqsubseteq_k^{k_E} \quad \iff \quad \neg_{sub} B \sqsubseteq_k^{k_E} \neg_{sub} A \tag{A.26}$$

when $k = 1$

Proof. We have:

$$A \sqsubseteq_1^{k_{\mathsf{E}}} B \quad \iff \quad E = 0 \tag{A.27}$$

But if $E = 0$ that means that $B - A$ is positive, i.e. no error term is required. But then by Theorem 5, we know that $\neg_{sub}\mathsf{A} - \neg_{sub}\mathsf{B}$ is positive. Thus $E' = 0$ when calculating:

$$\neg_{sub}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{E}}} \neg_{sub}\mathsf{A} \quad \iff \quad E' = 0 \tag{A.28}$$

Thus we have:

$$\mathsf{A} \sqsubseteq_1^{k_{\mathsf{E}}} \mathsf{B} \quad \iff \quad \neg_{sub}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{E}}} \neg_{sub}\mathsf{A} \tag{A.29}$$

□

k_{BA} - ✓ Lewis (2019) shows that the *subtraction-from-identity-negation* preserves the contrapositive for k_{BA}.

A.1.2.2 The support-inverse-negation

k_{hyp} - ✓

Theorem 7 (Rodatz et al., 2021)**.** *For two positive operators A and B with $rank(A) = rank(B)$, k_{hyp} is reversed under $\neg_{supp}$:*

$$A \sqsubseteq_k^{k_{hyp}} B \quad \iff \quad \neg_{supp} B \sqsubseteq_k^{k_{hyp}} \neg_{supp} A \tag{A.30}$$

Proof. From Baksalary et al. (1989, Theorem 4.3), $\neg_{supp}$ reverses Löwner order (i.e. k_{hyp} with $k = 1$) when $rank(\mathsf{A}) = rank(\mathsf{B})$, meaning

$$\mathsf{A} \sqsubseteq_1^{k_{\mathsf{hyp}}} \mathsf{B} \quad \iff \quad \neg_{supp}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{hyp}}} \neg_{supp}\mathsf{A} \tag{A.31}$$

Thus

$$\begin{aligned}
\mathsf{A} \sqsubseteq_k^{k_{\mathsf{hyp}}} \mathsf{B} &\iff \mathsf{B} - k\mathsf{A} \geq 0 && \text{(A.32)}\\
&\iff k\mathsf{A} \sqsubseteq_1^{k_{\mathsf{hyp}}} \mathsf{B} && \text{(A.33)}\\
&\iff \neg_{supp}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{hyp}}} \neg_{supp}(k\mathsf{A}) && \text{(A.34)}\\
&\iff \neg_{supp}(k\mathsf{A}) - \neg_{supp}\mathsf{B} \geq 0 && \text{(A.35)}\\
&\iff \frac{1}{k}(\neg_{supp}\mathsf{A}) - \neg_{supp}\mathsf{B} \geq 0 && \text{(A.36)}\\
&\iff \neg_{supp}(\mathsf{A}) - k(\neg_{supp}\mathsf{B}) \geq 0 && \text{(A.37)}\\
&\iff \neg_{supp}\mathsf{B} \sqsubseteq_k^{k_{\mathsf{hyp}}} \neg_{supp}\mathsf{A} && \text{(A.38)}
\end{aligned}$$

where we use "≥ 0" to denote that an operator is positive.

We use the fact that $k\mathsf{A}$ has the same spectral decomposition A with all eigenvalues are multiplied by k to get from Equation A.35 to Equation A.36.

□

k_{E} - ✗ For any two positive operators A, B, we have

$$\mathsf{A} \sqsubseteq_k^{k_{\mathsf{E}}} \mathsf{B} \text{ with } k = 1 - \frac{\|\mathsf{E}\|}{\|\mathsf{A}\|} \tag{A.39}$$

Similarly we have

$$\neg_{supp}\mathsf{B} \sqsubseteq_{k'}^{k_{\mathsf{E}}} \neg_{supp}\mathsf{A} \text{ with } k' = 1 - \frac{\|\mathsf{E}'\|}{\|\neg_{supp}\mathsf{B}\|} \tag{A.40}$$

Thus, even if $\mathsf{E} = \mathsf{E}'$, which is the case for A and B having the same eigenbasis, $\|\mathsf{A}\| \neq \|\neg_{supp}\mathsf{B}\|$. Thus the contrapositive does not hold.

k_{BA} - ✓

Theorem 8 (Rodatz et al., 2021)**.** *For two invertible positive operators A and B with the same eigenbasis, k_{BA} is reversed by matrix inverse:*

$$B^{-1} \sqsubseteq_{=k}^{k_{BA}} A^{-1} \quad \Longleftrightarrow \quad A \sqsubseteq_{=k}^{k_{BA}} B \tag{A.41}$$

Proof.

$$B^{-1} \sqsubseteq_{=k}^{k_{\mathsf{BA}}} A^{-1} \Longleftrightarrow k = \frac{\sum_i \lambda^i_{\mathsf{A}^{-1}} - \lambda^i_{\mathsf{B}^{-1}}}{\sum_i \left|\lambda^i_{\mathsf{A}^{-1}} - \lambda^i_{\mathsf{B}^{-1}}\right|} \tag{A.42}$$

$$= \frac{\sum_i \frac{1}{\lambda^i_{\mathsf{A}}} - \frac{1}{\lambda^i_{\mathsf{B}}}}{\sum_i \left|\frac{1}{\lambda^i_{\mathsf{A}}} - \frac{1}{\lambda^i_{\mathsf{B}}}\right|} \tag{A.43}$$

$$= \frac{\sum_i \lambda^i_{\mathsf{B}} - \lambda^i_{\mathsf{A}}}{\sum_i |\lambda^i_{\mathsf{B}} - \lambda^i_{\mathsf{A}}|} \tag{A.44}$$

$$\Longleftrightarrow \mathsf{A} \sqsubseteq_{=k}^{k_{\mathsf{BA}}} \mathsf{B} \tag{A.45}$$

using that for some invertible matrix X with spectral decomposition $\mathsf{X} = \sum_i \lambda_i |i\rangle \langle i|$, we have $\mathsf{X}^{-1} = \sum_i \frac{1}{\lambda_i} |i\rangle \langle i|$ to get from Equation A.42 to A.43. □

But then as for an invertible matrix $\neg_{supp}$ is equal to the matrix inverse, we have:

Corollary 1. *For two invertible positive operators A and B, with the same eigenbasis, k_{BA} is reversed by $\neg_{supp}$, i.e.:*

$$A \sqsubseteq_{=k}^{k_{BA}} B \quad \Longleftrightarrow \quad \neg_{supp} B \sqsubseteq_{=k}^{k_{BA}} \neg_{supp} A \tag{A.46}$$

A.1.2.3 The kernel-inverse-negation

k_{hyp} - ✓

Theorem 9. *For two positive operators A and B we have:*

$$A \sqsubseteq_{=k}^{k_{hyp}} B \quad \Longleftrightarrow \quad \neg_{ker} B \sqsubseteq_{=1}^{k_{hyp}} \neg_{ker} A \tag{A.47}$$

Please observe that on the right-hand side, we have $k' = 1$.

Proof. We have:

$$A \sqsubseteq_k^{k_{\text{hyp}}} B \iff supp(A) \subseteq supp(B) \tag{A.48}$$

$$\iff ker(B) \subseteq ker(A) \tag{A.49}$$

$$\iff supp(\neg_{ker} B) \subseteq supp(\neg_{ker} A) \tag{A.50}$$

But then, as we know that for any positive operator X, we know that $\neg_{ker} X$ is the identity over its support, we have:

$$A \sqsubseteq_k^{k_{\text{hyp}}} B \iff \neg_{ker}\mathsf{B} \sqsubseteq_1^{k_{\text{hyp}}} \neg_{ker}\mathsf{A} \tag{A.51}$$

□

Corollary 2. *For two invertible density matrices A and B, the contrapositive is observed for crisp Löwner order. We have:*

$$\mathsf{A} \sqsubseteq_1^{k_{hyp}} \mathsf{B} \iff \neg_{ker}\mathsf{B} \sqsubseteq_1^{k_{hyp}} \neg_{ker}\mathsf{A} \tag{A.52}$$

k_{E} - ✓

Theorem 10. *For two positive operators A and B, we have:*

$$\mathsf{A} \sqsubseteq_1^{k_E} \mathsf{B} \iff \neg_{ker}\mathsf{B} \sqsubseteq_1^{k_E} \neg_{ker}\mathsf{A} \tag{A.53}$$

Proof. We have:

$$A \sqsubseteq_1^{k_{\mathsf{E}}} B \iff E = 0 \tag{A.54}$$

But if $E = 0$ that means that $B - A$ is positive, i.e. no error term is required. But then by Corollary 2, we know that $\neg_{ker}\mathsf{A} - \neg_{ker}\mathsf{B}$ is positive. Thus $E' = 0$ when calculating:

$$\neg_{ker}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{E}}} \neg_{ker}\mathsf{A} \iff E' = 0 \tag{A.55}$$

Thus we have:

$$\mathsf{A} \sqsubseteq_1^{k_{\mathsf{E}}} \mathsf{B} \iff \neg_{ker}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{E}}} \neg_{ker}\mathsf{A} \tag{A.56}$$

□

k_{BA} - ✓

Theorem 11. *For two positive operators A and B, we have:*

$$\mathsf{A} \sqsubseteq_1^{k_{BA}} \mathsf{B} \iff \neg_{ker}\mathsf{B} \sqsubseteq_1^{k_{BA}} \neg_{ker}\mathsf{A} \tag{A.57}$$

Proof. We have:

$$A \sqsubseteq_1^{k_{\mathsf{BA}}} B \iff E = 0 \tag{A.58}$$

But if $E = 0$ that means that $B - A$ is positive, i.e. no error term is required. But then by Corollary 2, we know that $\neg_{ker}\mathsf{A} - \neg_{ker}\mathsf{B}$ is positive. Thus $E' = 0$ when calculating:

$$\neg_{ker}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{BA}}} \neg_{ker}\mathsf{A} \iff E' = 0 \tag{A.59}$$

Thus we have:

$$\mathsf{A} \sqsubseteq_1^{k_{\mathsf{BA}}} \mathsf{B} \iff \neg_{ker}\mathsf{B} \sqsubseteq_1^{k_{\mathsf{BA}}} \neg_{ker}\mathsf{A} \tag{A.60}$$

□

A.1.2.4 The inverse-negation

k_{hyp} - ✗ Despite have some desirable interaction with k_{hyp} for both $\neg_{supp}$ and $\neg_{ker}$, $\neg_{inv}$ does not interact well with k_{hyp}. This is due to the normalisation of the *support-inverse-negation* before taking the sum. We can take for example the following two matrices with the same rank:

$$\mathsf{A} = \begin{pmatrix} 0.1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{A.61}$$

$$\mathsf{B} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{A.62}$$

Then $\mathsf{A} \sqsubseteq_1^{k_{\mathsf{hyp}}} \mathsf{B}$ when considering k_{hyp} as we have:

$$\mathsf{B} - \mathsf{A} = \begin{pmatrix} 0.9 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \geq 0 \tag{A.63}$$

But we have

$$\neg_{inv}\mathsf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \text{see example in A.1.1.4)} \tag{A.64}$$

$$\neg_{inv}\mathsf{B} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{A.65}$$

Therefore we only have $\neg_{inv}\mathsf{B} \sqsubseteq_{0.1}^{k_{\mathsf{hyp}}} \neg_{inv}\mathsf{A}$ as for any higher k we would have a negative eigenvalue for the second eigenvector. Thus for this simple example, we have shown that the contrapositive does not hold.

k_{E} - ❌ The contrapositive does not hold for $\neg_{supp}$. Thus for all positive operators, that have an empty kernel, the contrapositive also does not hold for $\neg_{inv}$. Thus $\neg_{inv}$ does not obey the contrapositive.

k_{BA} - ❌ While the contrapositive may hold for $\neg_{supp}$ and $\neg_{ker}$ under certain conditions, it does not hold for $\neg_{inv}$. In particular, even if both positive operators are invertible, the normalisation after taking the *support-inverse-negation* destroys the contrapositive. This is similar to the example for the contrapositive for k_{hyp}.

B
Additional Data

B.1 Framework comparison

The results presented in Section 4.5.3 are for the frameworks under the context function $\texttt{hyp-khyp}_4$. Under that function we get an optimal value of 0.654 for CN_{word1} and CN_{word2}. CN_{word3} and CN_{word4} perform optimally under $\texttt{poly}_4$. The results are presented in Table B.1. The scores in this table correspond to the scores presented in Rodatz et al. (2021)[1]

One key observation is that under this function, more frameworks perform well, yet none as good as the optimal framework under $\texttt{hyp-khyp}_4$.

[1]Please observe that these scores slightly differ from the scores presented in Rodatz et al. (2021). This difference is due to minor changes in the implementation for missing data and rounding. The differences are not significant.

Table B.1: Pearson correlation of different framework under context function $\texttt{poly}_4$ with human intuition. Correlations above 0.4 are highlighted in green.

Framework	Logical negation	Composition	k_{E1}	k_{E2}	k_{hyp1}	k_{hyp2}	k_{BA}	sim_{trace}
w_N	—	—	0.464	0.551	0.303	-0.003	0.268	0.575
$\texttt{wc}_{w_N}$	—	—	0.440	0.599	0.294	0.475	0.318	0.628
CN_{word1} & CN_{word2} *(negations give same results under these composition operations)*	$\neg_{sub}$	spider_{w}	-0.190	-0.220	0.278	0.231	0.269	-0.094
		phaser_{w}	0.414	0.604	0.302	0.493	0.303	0.627
		fuzz_{w}	-0.232	-0.074	0.293	0.241	0.271	0.460
		diag	-0.260	-0.234	0.293	0.028	0.269	-0.038
	$\neg_{supp}$	spider_{w}	0.176	0.400	0.259	-0.072	0.182	0.385
		phaser_{w}	-0.147	0.146	0.250	0.076	0.148	0.180
		fuzz_{w}	-0.178	0.064	0.256	0.044	0.146	0.021
		diag	-0.236	0.028	0.173	0.053	0.121	-0.048
	$\neg_{ker}$	spider_{w}	-0.253	-0.246	0.116	0.110	0.172	-0.459
		phaser_{w}	0.354	0.459	0.306	0.298	0.213	0.555
		fuzz_{w}	-0.223	-0.075	0.296	0.097	0.183	0.294
		diag	-0.243	-0.172	0.294	-0.003	0.180	0.040
	$\neg_{inv}$	spider_{w}	-0.159	-0.016	0.250	0.081	0.149	0.127
		phaser_{w}	0.343	0.494	0.309	0.230	0.215	0.566
		fuzz_{w}	-0.223	-0.088	0.296	0.04	0.195	0.255
		diag	-0.253	-0.201	0.290	-0.007	0.196	0.009
CN_{word1}	$\neg_{sub}$	spider_{c}	-0.067	0.271	0.290	0.505	0.256	0.526
		phaser_{c}	-0.270	-0.320	0.296	-0.269	0.275	-0.302
		fuzz_{c}	-0.262	-0.207	0.296	0.016	0.276	-0.067
	$\neg_{supp}$	spider_{c}	0.258	0.397	0.250	0.265	0.165	0.458
		phaser_{c}	-0.106	-0.001	0.235	0.071	0.148	0.268
		fuzz_{c}	-0.218	-0.015	0.230	0.057	0.145	0.046
	$\neg_{ker}$	spider_{c}	-0.086	0.183	0.268	0.383	0.180	0.523
		phaser_{c}	-0.271	-0.290	0.291	-0.202	0.204	-0.323
		fuzz_{c}	-0.239	-0.199	0.291	-0.023	0.204	-0.075
	$\neg_{inv}$	spider_{c}	0.053	0.306	0.281	0.385	0.152	0.475
		phaser_{c}	-0.266	-0.234	0.294	-0.102	0.197	-0.173
		fuzz_{c}	-0.228	-0.190	0.295	-0.012	0.196	-0.059
CN_{word2}	$\neg_{sub}$	spider_{c}	-0.152	0.068	0.282	0.366	0.241	0.322
		phaser_{c}	-0.270	-0.279	0.303	0.151	0.259	-0.262
		fuzz_{c}	-0.252	-0.125	0.302	0.143	0.261	-0.01
	$\neg_{supp}$	spider_{c}	0.256	0.417	0.261	0.057	0.174	0.429
		phaser_{c}	-0.069	0.032	0.245	-0.287	0.148	0.282
		fuzz_{c}	-0.193	-0.016	0.229	0.036	0.136	0.088
	$\neg_{ker}$	spider_{c}	-0.173	0.031	0.267	0.215	0.170	0.252
		phaser_{c}	-0.254	-0.259	0.306	0.100	0.187	-0.283
		fuzz_{c}	-0.212	-0.135	0.303	0.112	0.188	-0.018
	$\neg_{inv}$	spider_{c}	-0.014	0.247	0.275	0.342	0.140	0.375
		phaser_{c}	-0.256	-0.188	0.302	0.059	0.180	-0.16
		fuzz_{c}	-0.176	-0.127	0.303	0.117	0.179	-0.016
CN_{word3}	—	—	0.237	0.477	0.305	0.388	0.241	0.598
CN_{word4}	—	—	0.240	0.424	0.306	0.257	0.224	0.583