Functorial Language Models

Alexis Toumi[1] Alex Koziell-Pipe[2]

[1]University of Oxford, Cambridge Quantum
[2]University of Oxford (from October 2021)

15 July
ACT 2021

Outline

- BERT, GPT-3 have made strides toward end-to-end NLP.
- Interpretability is still a challenge.
- DisCoCat models give some hope in opening the black box.
- Can we integrate DisCoCat models into an end-to-end framework?
- First steps towards this via a functorial approach.
    - *Functorial Learning*
- Application to missing word prediction.
- *Functorial Language Models.*

Language Models

- *Language Model:* probability distribution over word sequences

$$w_1 w_2 \ldots w_n \longmapsto P(w_1 w_2 \ldots w_n)$$

- Extensive use in state of the art NLP (BERT, GPT-3) end-to-end.
- Interpretability is still a challenge; when we open up these models, we're still just looking at matrices of weights...
- A language model based on DisCoCat could improve on this by adding an explicit interpretation of grammatical structure, *categorically*.

Pregroup Category

Pregroup Grammar

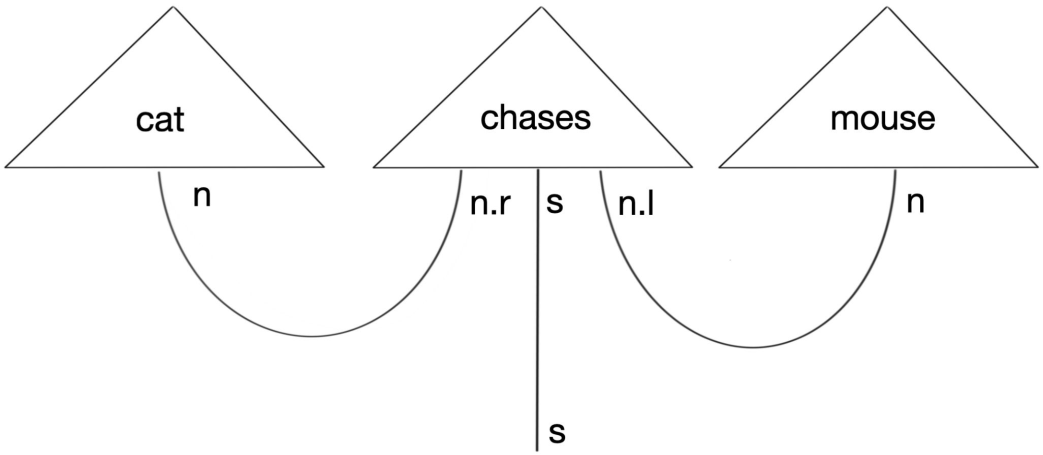
$$G = (V, X, R, s)$$

Vocabulary	V
Grammatical types	X
Grammatical Rules	R
Sentence type	s

Defines a rigid monoidal category:

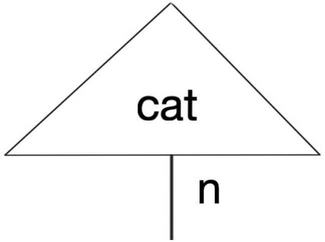
- Objects generated by $V + X$
- Morphisms generated by R

Grammatical Derivations ↔ String Diagrams



For pregroups, rules in R are dictionary entries of the form

$$w \rightarrow t$$



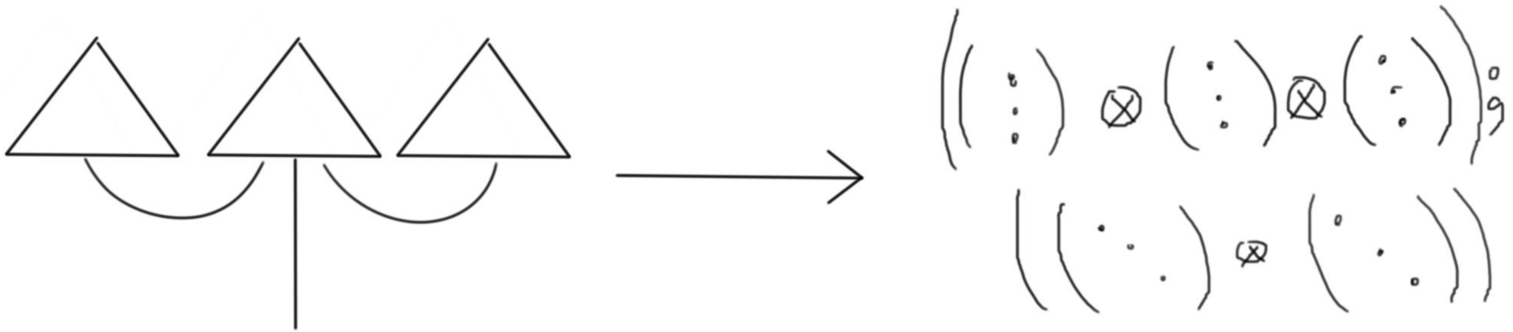
(as well as cups)

DisCoCat

A DisCoCat model is a monoidal functor $F : \mathbf{G} \to \mathbf{S}$, where

- ***G*** is a grammar category
  - grammatical types as objects
  - grammatical reductions as morphisms
- ***S*** is a semantic category
  - e.g. **FVect**, CPM(**FVect**), ...

Example: **Pregroup → FVect**



DisCoCat End-to-End

- Despite some empirical validation on small datasets, DisCoCat is yet to be applied at scale.

$$\text{Raw Text} \longrightarrow \underset{\mathbf{G}}{\text{Grammar}} \xrightarrow[]{\text{DisCoCat } F} \underset{\mathbf{S}}{\text{Semantics}}$$

- Two-fold challenge:
    - Predicting the grammar, given a word sequence.
    - Learning the representation of word meanings.

DisCoCat End-to-End

- Despite some empirical validation on small datasets, DisCoCat is yet to be applied at scale.

- Two-fold challenge:
  - Predicting the grammar, given a word sequence.
  - Learning the representation of word meanings.

Functorial Language Models

- Let F : **Pregroup** $\rightarrow$ **FVect** be a DisCoCat model. Can we describe its action on objects and morphisms in a concise way?
  - Objects: a map from grammatical types to natural numbers

$$F_0 : X \rightarrow \mathbb{N}$$

  - Morphisms: a set of maps from vocabulary to vectors*

$$F_1 = \left\{ V_t \rightarrow \mathbb{R}^{F(t)} \,\middle|\, t \in G_0 \right\}$$

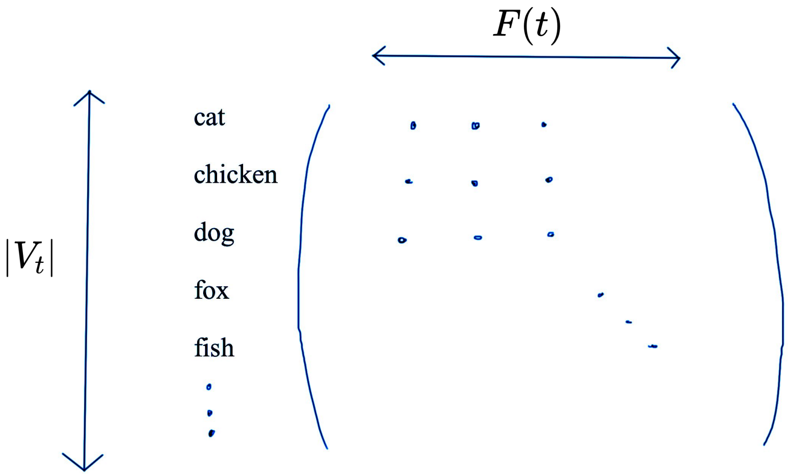
- Claim: *we can encode all the information about this functor within a set of matrices.*
  - *"Encoding Matrices"*
- …treat the matrix entries as parameters, and *learn the functor from data*.
  - *"Functorial Learning"*

$${}^*V_t = \{ w \mid w \in V, w \rightarrow t \in R \}$$

Encoding Matrices

- Order the words in our vocabulary according to some canonical (e.g. alphabetical) order.
- For a set V_t * of vocabulary words of grammatical type $t \in G_0$, define an encoding matrix:

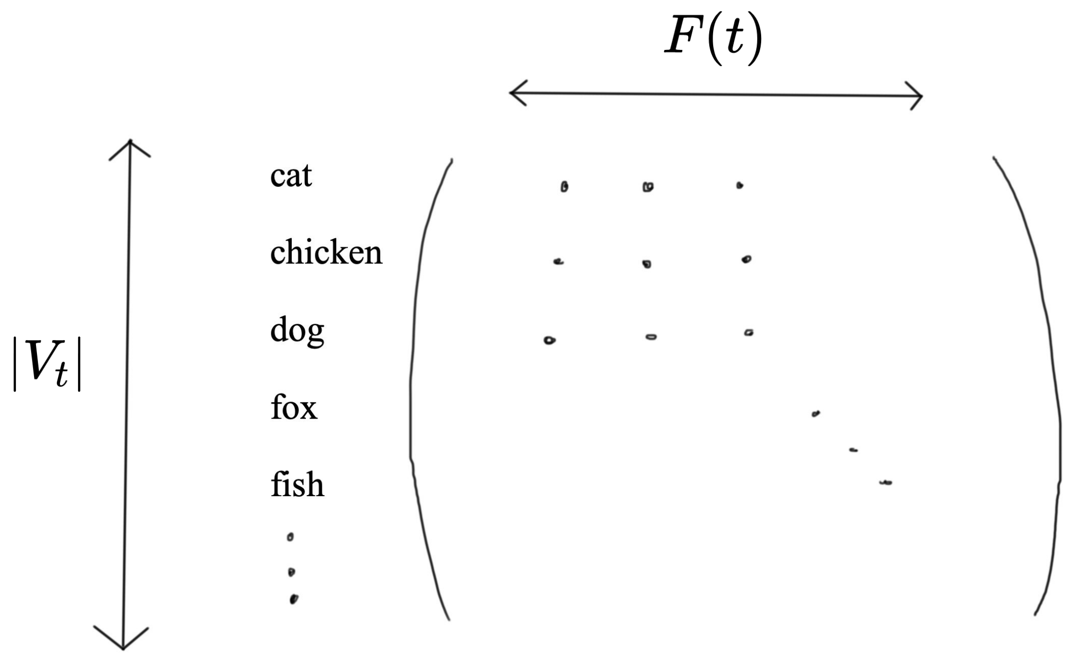
$$E_t : |V_t| \times F(t) \to \mathbb{R}$$



- The object mapping $F_0 : X \to \mathbb{N}$ is given by the "widths" of the matrices, and can be considered a set of hyperparameters of the model.

$^*V_t = \{ w \mid w \in V,\, w \to t \in R \}$

- Each row corresponds to the vector mapped for a certain word in the vocabulary.

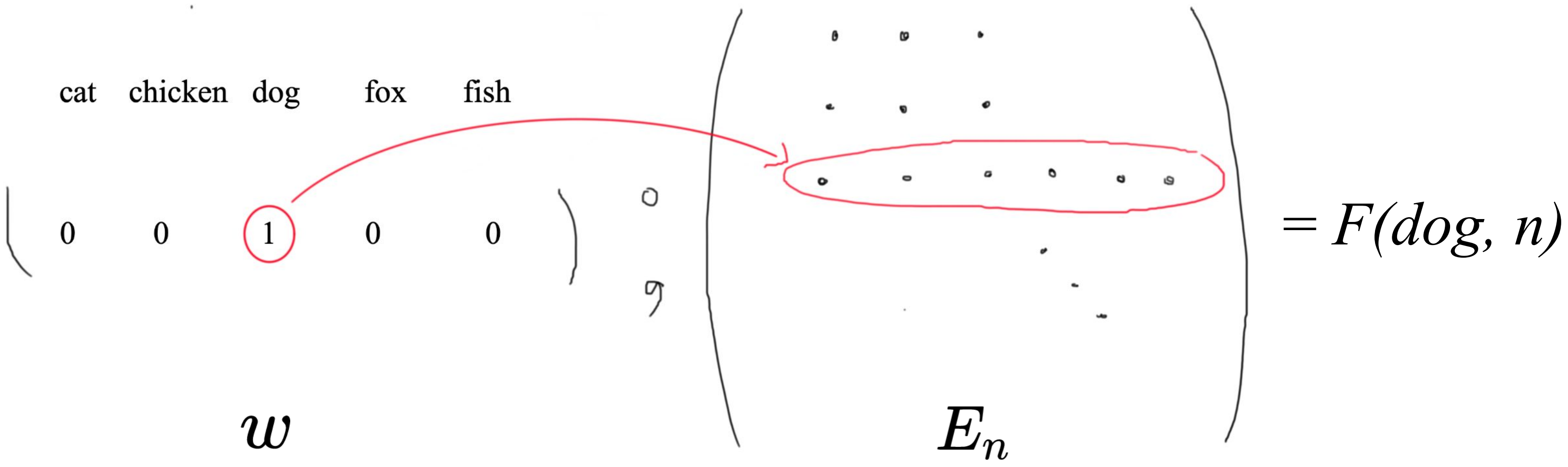


$$E_t : |V_t| \times F(t) \to \mathbb{R}$$

Encoding Matrices

- Each row corresponds to the vector mapped to from a certain word in the vocabulary.
- Hence the image of the functor F on a word can be obtained via composition (pre-multiplication) with a one-hot vector w:

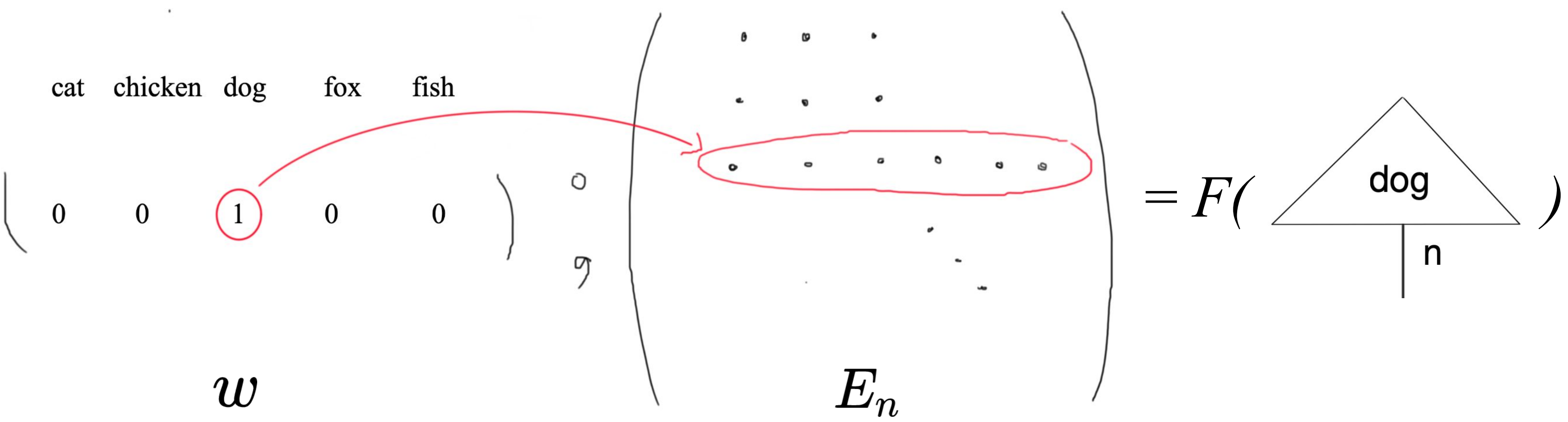
$$w \fatsemi E_t = F(w, t)$$



Encoding Matrices

- Each row corresponds to the vector mapped to from a certain word in the vocabulary.
- Hence the image of the functor *F* on a word can be obtained via composition (pre-multiplication) with a one-hot vector w:

$$w \fatsemi E_t = F(w, t)$$



(Supervised) Functorial Learning

- Given a dataset $X \subseteq Ar(\mathbf{G}) \times Ar(\mathbf{S})$, compute (or approximate):

$$F^{\star} = \underset{F\,:\,\mathbf{G} \to \mathbf{S}}{argmin} \left(\; R(F_1) + \sum_{(d,y) \in X} L(F(d),\, y) \; \right)$$

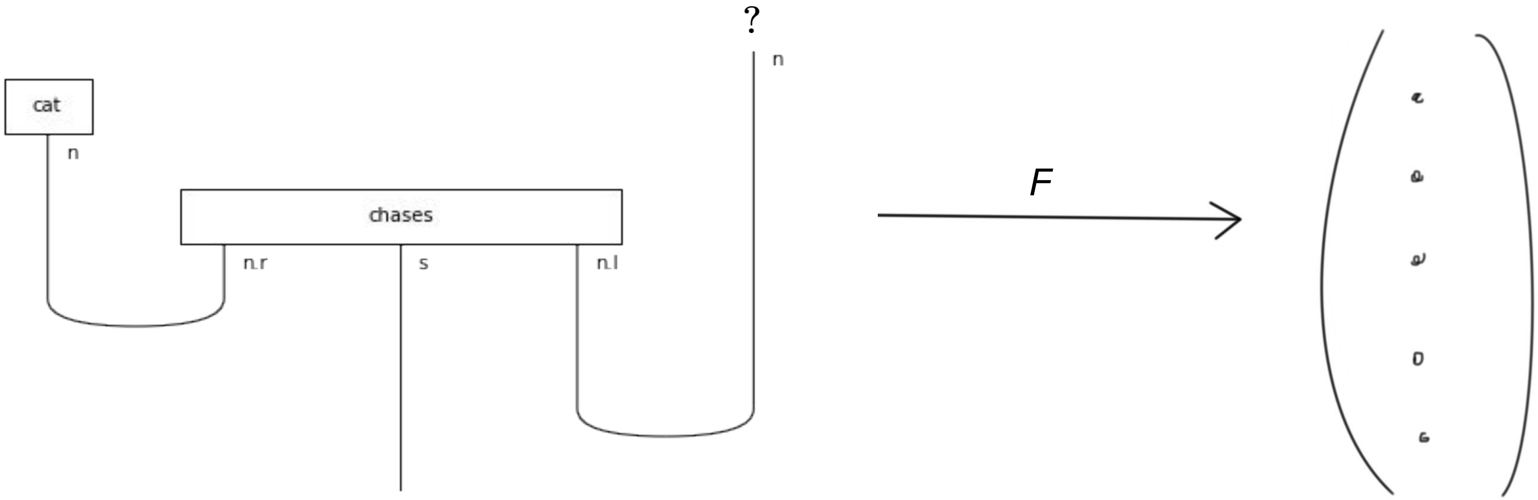
- Where
  - R is a regularization over the mapping on morphisms (encoding matrices).
  - L is a loss function appropriate to the learning task.

- Fix $F(s) = 1$. Then the value of $F(d)$ turns out as a scalar, and could be thought of as the "truth or false-ness" of a sentence. Labels $y \in \{0,\, 1\}$ could be used to simulate question-answering[1,2]

[1] de Felice, Meichanetzidis, & Toumi, *Functorial Question Answering (2019)*
[2] Meichanetzidis, Toumi, de Felice, & Coecke, *Grammar-Aware Question-Answering on Quantum Computers (2020)*

Experiment

- Randomly initialize the functor (encoding matrices).
- Remove a word (box) from a valid sentence, use the functor to map diagram to vector.

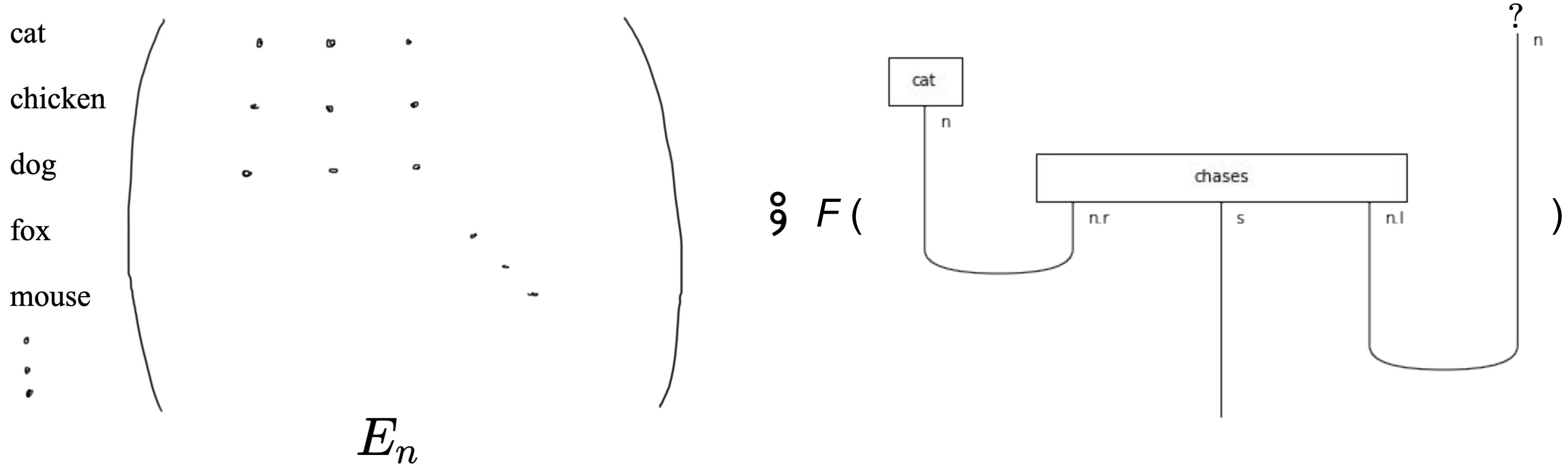


"DisCoPy: Monoidal Categories in Python." arXiv:2005.02975

github.com/oxford-quantum-group/discopy/

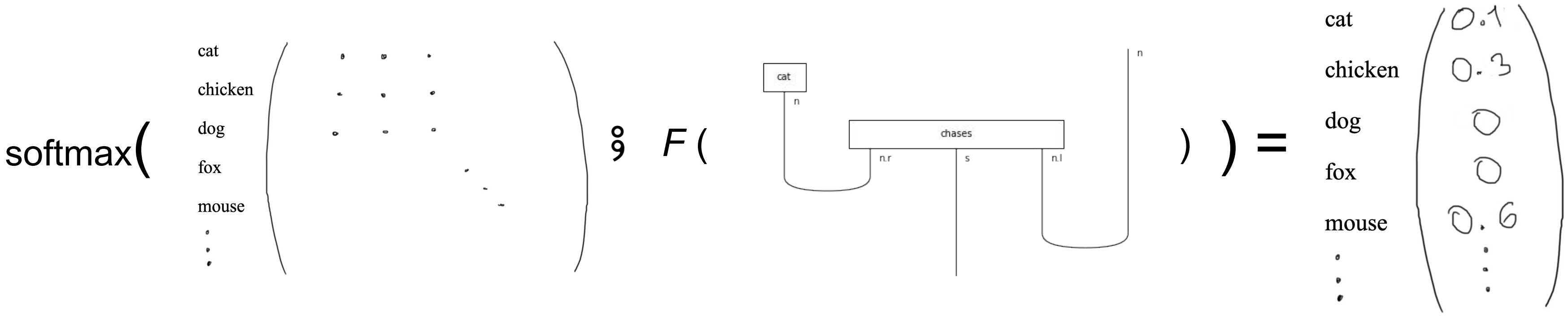
Experiment

- Precompose with encoding matrix of missing word type.

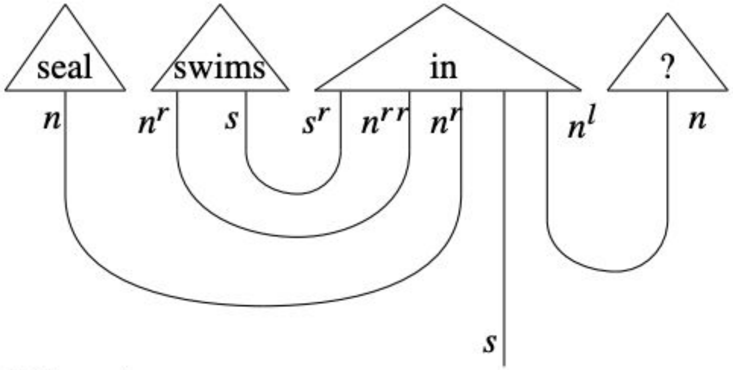


Experiment

- Apply softmax function $\sigma(\theta_i) = \frac{e^{\theta i}}{\sum_i e^{\theta i}}$ to obtain a probability distribution over the vocabulary.
- Compare to a ground truth label, and update the functor via gradient-based methods.

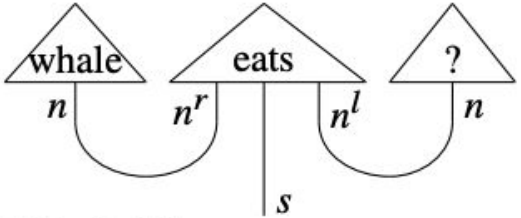


Experiment



Target: water

Prediction: water (0.97), dog (0.02)



Target: krill

Prediction: cheese (0.53), fish (0.17), grain (0.10)

Future

- Combine with a probabilistic grammar $P(d|w_1,\ldots,w_n)$[1]
- Use a trained model to generate sentences[2], towards generative adversarial
- Use "bubbles" to encode softmax[3]
- Learn the functor in an unsupervised manner
- Upscaling to larger datasets
- Vary the grammar and semantic categories.
- Replace "encoding matrices" with "encoding networks"
- Make it quantum by considering functors **G** → **QCirc**

---

[1] Schiebler, Toumi & Sadrzadeh, *Incremental Monoidal Grammars (2020)*
[2] de Felice, Di Lavore, Román & Toumi, *Functorial Language Games for Question Answering (2020)*
[3] Toumi, Yeung, & de Felice, *Diagrammatic Differentiation for Quantum Machine Learning (2021)*

Thank you!