Entangled Meanings: Classification and Ambiguity Resolution in QNLP

Chi Zhang
Department of Mathematics
Indiana University
Bloomington, IN, USA
czh4@iu.edu

Akriti Kumari
Luddy School of Informatics, Computing, and Engineering
Indiana University
Bloomington, IN, USA
akkumari@iu.edu

Damir Cavar
NLP-Lab
Indiana University
Bloomington, IN, USA
dcavar@iu.edu

***Abstract*—We discuss experiments involving two tasks in Quantum Natural Language Processing (QNLP): text classification and disambiguation. In the classification task, we utilized an amplitude encoding algorithm and achieved perfect accuracy on the `lambeq` dataset discussed in literature. We obtained accuracy from 55% to 72.5% on the more complex and realistic Amazon review dataset. This is a reasonable result given the current state-of-the-art results in QNLP. Additionally, when using vector dimension reduction for embeddings, we found that UMAP leads to the best results in our experiment setting. All classification results were done on the `default.qubit` simulator in `pennylane 0.36` python library. Our classification results highlight the potential of quantum algorithms in practical applications. In the disambiguation task, we selected 18 ambiguous nouns, 32 unambiguous nouns, and 18 different verbs. Our experiments using the `QASM simulator` within the `qiskit` Python library demonstrated that the simulator could perfectly differentiate between the various meanings of ambiguous nouns in different contexts. Furthermore, we extended our study to a real quantum device, the `ibm_kyoto` quantum computer. There, we tested our disambiguation approach on a subset of 4 random nouns (2 ambiguous and 2 unambiguous) and observed that `ibm_kyoto` could achieve an accuracy range of 82.1% to 98.9% in disambiguation tasks, extending the datasets and improving the results of existing ambiguity resolution experiments [1]. Our work demonstrates the capability of quantum computing in dealing with real-world NLP tasks, hence contributing to the advancement of both QML and NLP.**

***Index Terms*—Natural Language Processing, Semantics, Sentiment Analysis, Supervised Machine Learning**

I. Introduction

Quantum computing algorithms have emgerged since the 1990s, with pioneering work by researchers like Shor and Grover [2], [3]. Quantum computing has shown achievements in various disciplines in recent years, like chemistry [4], finance [5], machine learning [6], optimization [7] and cryptography [8].

Quantum Natural Language Processing (QNLP) is such a particularly promising subfield of Quantum Machine Learning (QML) [9]. QNLP is aimed to leverage the power of quantum computers to perform Natural Language Processing (NLP) tasks, potentially demonstrating a quantum advantage in such NLP tasks. A potential quantum advantage in QNLP could be related to vector spaces being a successful model to describe natural language properties (e.g., semantics and to some extend syntax) and at the same time vector spaces being an inherent property of quantum computing [10]. The compatibility between vector space models and quantum computing is rooted in the similarity of the mathematical structures of both quantum mechanics and state-of-the-art models of natural language syntax and semantics. In quantum mechanics, vector spaces are physical Hilbert spaces of quantum states [11]. On the other hand, linguists have long used vector spaces to model linguistic semantic spaces [12]. It is astonishing that there is an apparent morphism between the vector spaces in these two seemingly unrelated disciplines. In some QNLP work this is shown using Lambek's pregroup grammar [13], [14] as a model for natural language syntax. The similarity between quantum mechanics and natural language suggests a potential natural fit for QNLP on quantum hardware [10]. This emerging field has drawn increasing attention due to its potential to understand how quantum machines understand and process natural language [15].

Within QNLP, we focus on two schools of thought, among various other approaches. The first approach, as mentioned earlier, is based on the similarity between quantum mechanics and natural language, utilizing mathematical frameworks like pre-group grammar [13] and category theory [16] to model some linguistic properties. Foundational frameworks in this area have been established by authors like Bob Coecke in the 2010s [14], [17]–[19]. These frameworks integrate syntax and semantic natural language properties in quantum mechanics via pregroup grammar, a formalism with context-sensitive complexity [14], [20] that captures the syntactic structure of natural language in a mathematically rigorous way. This line of research has led to the development of systems like DisCoCat [14] and DisCoPy [21] that have become the groundwork of the first QNLP experiment on IBM quantum computers [22]. The DisCoCat and DisCoPy frameworks have bridged the gap between linguistic theories and quantum mechanics.

The second approach to QNLP is based on Machine Learning algorithms. Authors including Widdows [1] have demonstrated the potential of state-of-the-art quantum computers for processing real-life language datasets. They conducted a classification task on realistic IMDB dataset [23], highlighting the feasibility of QNLP applications. Additionally, they performed the first QNLP ambiguity resolution task on an 11-qubit IonQ

979-8-3315-4137-8/24/$31.00 ©2024 IEEE
DOI 10.1109/QCE60285.2024.10260

quantum computer. However, due to the limited size of the IonQ computer, their dataset was small, having only two verbs and five nouns. Despite these limitations, their work presents a significant step in demonstrating practical applications of QNLP and quantum computing.

In this work, we aim to further demonstrate the capabilities of quantum computers in dealing with classification and ambiguity resolution tasks. Building on the methodology and results from previous research [1], we explore algorithms in QML, trying to improve both the performance and the scope of QNLP applications. Through this research, we also seek to contribute to the growing QNLP community by providing open-source code, finally enhancing our knowledge of how quantum computers can understand and process real-life human language.

II. Approaches

A. Classification

We conducted binary classification experiments on datasets from the `lambeq` [20] and the Amazon review dataset [24].

We first vectorized the texts in the datasets using the `Word2Vec` [25] word embeddings in the `gensim 4.1.2` Python library [26], as well as the `spaCy 3.7` library [27]. The `Word2vec` vectors serve as a baseline and are not state-of-the-art embedding models. The vector length in `Word2vec` for every word token is 300 dimensions of real numbers.

To reduce the computational effort, we applied various dimension reduction techniques to the `Word2vec` word embeddings. The dimension reduction techniques utilized in this paper are PCA, LDA, and t-SNE from the `scikit-learn` library [28], and UMAP from the `umap-learn` library [29] for manifold learning. Following this, we implement the amplitude encoding algorithm and divide–and–conquer encoding algorithm from [30], using the `qml` module from the Python package `Pennylane 0.37.0` [31].

Finally, we train our pipelines on the `default.qubit` simulator from `pennylane` at 15 iterations. Each iteration was comprising 150 steps and a learning rate of 0.1. The data was split into train, validation,and test sets with a ratio of 6:2:2. Subsequently, we record the training accuracy, validation accuracy, and test accuracy for each combination of {dataset, vectorizer, dimension reduction method, quantum encoding algorithm} , to identify the most economic combinations for scenarios with limited computing resources.

B. Ambiguity Resolution

Natural language words and utterances can be ambiguous, and the capacity to disambiguate linguistic expressions in specific contexts and situations is a fundamental property of the human language faculty [32]. Disambiguation in NLP represents one of the early successes of word vector models via term-document or term-term co-occurrence matrices [33]. While in current NLP approaches all words are represented the same, i.e., vectors of the same length, in early NLP proposals preceding the Deep Learning revolution, nouns were for example treated as vectors in semantic space, and adjectives modifying nouns were treated as matrices acting on the vectors in computational linguistics [34]. In newer NLP approaches in QNLP, authors like Widdows [1] treated nouns as vectors and transitive verbs as matrices.

Before developing the quantum framework that encodes verbs and nouns in quantum circuits, it's worth mentioning a simple linguistic insight. A common linguistic assumption is that predicates require arguments, e.g., noun phrases as their objects and subject, composing this way a complex predicate (Verb Phrase) or sentence. They could be assumed to be functions that map nominal arguments to complex predicates or sentences. Assuming that predicate heads like verbs (or nouns and adjectives) has led to interesting approaches using categorial grammar approaches in linguistics. From the early Bar–Hillel's work [35] in the 1960s, to the Combinatory Categorial Grammer (CCG) [36], or the Lambek's pregroup grammar [13], the influence of this simple insight can be seen in all of them.

Inspired by the aforementioned authors and insight, we treat nouns as vectors and transitive verbs as matrices in our computational model. Using vectors to record the information of nouns in the meaning space is a natural and common way [1], [32], [34]. If we can also use vectors to record information about verb phrases, we can then naturally formulate verbs as matrices, which are a common way to describe linear functions between two different vector spaces.

While the dimensionality of word embeddings in models like `Word2vec` is driven by empirical experimentation with real NLP tasks using neural models, in this approach, we determine the dimensions of noun vectors for classification purely for the classification task as such. To determine the dimension of the noun vectors, we classify these nouns into 16 distinct categories based on their meanings:

```
Animals, Technology/Devices, Food,
Body Parts, Nature, Plants, Tools/Equipment,
Music/Sound, Communication/Writing, Emotion/Feeling,
Weather/Elements, Action/Movement, Objects/Containers,
Traffic/Transportation, Espionage, People.
```

Therefore, we can represent both ambiguous and unambiguous nouns in our dataset as 16-dimensional vectors. For illustration, we list the vector representations of three ambiguous nouns—*Fan*, *Mole*, and *Date*—alongside three unambiguous nouns—*Screwdriver*, *Dolphin*, and *Infiltration* in Table I. This approach allows us to handle the complexity of ambiguous nouns while maintaining consistency in representation across the dataset.

To represent transitive verbs as matrices, we need to conceptualize them as functions that take for example nouns as input and produce verb phrases or complex predicates as output. To determine the appropriate dimensions, we must classify all meaningful output verb phrases, i.e., combinations of transitive verbs and nouns as objects. For example, in the case of the verb *cultivate*, the phrase *cultivate palm* is meaningful, whereas the phrase *cultivate ocean* is not. After thorough analysis,

	Fan	Mole	Date	Screwdriver	Dolphin	Infiltration
Animals	0	1	0	0	1	0
Tech/Device	1	0	0	0	0	0
Food	0	0	1	0	0	0
Body Parts	0	0	0	0	0	0
Nature	0	0	0	0	0	0
Plants	0	0	0	0	0	0
Tools/Equip	0	0	0	1	0	0
Music/Sound	0	0	0	0	0	0
Comm/Writing	0	0	1	0	0	0
Emotion/Feeling	0	0	0	0	0	0
Weather/Elements	0	0	0	0	0	0
Action/Movement	0	0	0	0	0	0
Objects/Containers	0	0	0	0	0	0
Traffic/Transport	0	0	0	0	0	0
Espionage	0	1	0	0	0	1
People	1	0	0	0	0	0

TABLE I
VECTOR REPRESENTATIONS OF *Fan*, *Mole*, *Date*, *Screwdriver*, *Dolphin* AND *Infiltration*

we classified all meaningful verb phrases into 10 distinct categories as follows:

Action, Operation, Consumption, Medical Examination, Exploration, Gardening, Communication, Emotion, Observation, Investigation.

Thus we can assume that each transitive verb is a function from a 16-dimensional vector space to a 10-dimensional vector space. We represent each verb as a 10×16 matrix, mapping the meaning space of nouns to the meaning space of verb phrases. For example, the verb *admire* can be represented by the 10×16 matrix in Table II, while the matrix of the verb *cultivate* is given in Table III. The entries in the matrices

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE II
MATRIX REPRESENTATION OF THE VERB *Admire*

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE III
MATRIX REPRESENTATION OF THE VERB *Cultivate*

capture the idea that the verbs take nouns as arguments and return a verb phrase between the column categories and row categories of the nonzero entries. E.g., the verb *cultivate* has only 1 nonzero entry in position $(6, 7)$, while the 6th category of nouns is *Plant* and the 7th category of verb phrases is *gardening*. Then we know that the verb *cultivate* is something that expects a plant name and give the meaning of a gardening activity. Similarly, we know that the verb *admire* is something that expects a person and expresses a particular feeling or emotion as shown in

We can convert the vectors and matrices into quantum circuits by applying X and CX gates to qubits representing each category, at the nonzero entries in the matrix and vector representations of the nouns and verbs. For example, the quantum circuit of the noun *Fan* is provided in Figure 1.

The quantum circuit for *admire* is provided in Figure 2.

With the quantum circuits of verbs and nouns, we can compose the circuits together into larger quantum circuits that represent the output verb phrases. To perform the ambiguity resolution task, we need to measure the output of the target dimension of the matrix to an ancillary classical bit. If the result of the measurement is '1', we know that the quantum computer can distinguish the difference between two meanings of the ambiguous noun, otherwise it fails to disambiguate the two meanings. For illustration, the quantum circuit for measuring the meaning of the verb phrase is *admire fan* is in Figure 3.

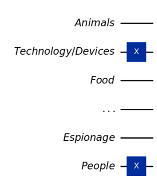

Fig. 1. Quantum Circuit for *Fan*

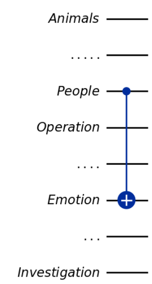

Fig. 2. Quantum Circuit for *Admire*

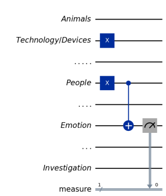

Fig. 3. Quantum Circuit for phrase *Admire Fan*

III. DATA AND RESULTS

A. Classification

The datasets can be seen in our Github repository.

We have listed our classification results with test accuracy greater than 53% in Table IV, with test accuracy greater than 80% highlighted.

We conclude that we have achieved perfect test accuracy on the `lambeq` dataset and accuracy 55% to 72.5% on the Amazon review dataset.

We also observe that on the `lambeq` dataset, both the combination {spaCy, UMAP, amplitude encoding} and {spaCy, None, amplitude encoding} (indicating no dimensionality reduction on the word vectors) achieve 100% test accuracy. However, the former combination requires only 1 qubit, whereas the latter uses 16 qubits to achieve the classification results. In contrast, with the Amazon review dataset, the combination {Word2Vec, UMAP, divide-and-conquer encoding} achieves the highest test accuracy in our experiments. Based on these observations, we conclude that UMAP is the most effective dimensionality reduction technique in our setting. We are aware of issues with the `Word2vec` word embeddings, e.g., the lack of disambiguation and conflation of multiple semantic properties in one word vector. However, this did not impact our experimental results.

What makes our results interesting is that we used only 1 qubit to achieve perfect accuracy on the `lambeq` dataset by combining UMAP and quantum encoding algorithms, whereas other researchers required 5 qubits in [22] and 4 qubits in [1]. This demonstrates a significant advancement in the field of QNLP experiments, particularly in scenarios with limited computing resources.

Due to the complex nature of natural language and online written reviews, our test accuracy on the Amazon review dataset ranges from 55% to 72.5%, indicating that current QML methods are still not fully equipped to process natural language. However, when compared to the 57% to 62% accuracy achieved on the IMDB dataset [23], our results are still reasonable.

B. Ambiguity Resolution

In the ambiguity resolution task our dataset consists of 50 nouns and 18 transitive verbs in English. Among the 50 nouns, 18 of them are ambiguous, exhibiting multiple meanings or senses depending on the context in which they are used.

```
Chest, Wave, Pitch, Jam,
Bark, Palm, Stress, Bat, Rock, Fan, Seal,
Duck, Mouse, Bolt, Mole, Date, Pen, Crane,
```

32 of the verbs are unambiguous, each verb we assumed to have a single main meaning.

```
Screwdriver, Infiltration, Dolphin, Book, Liver,
Rainbow, Thunderstorm, Bread, Email, Box, Symphony,
Tree, Flower, Computer, Sadness, Car, Heart, Guitar,
Air conditioner, Happiness, Ocean, Elephant,
Dance, Apple, Jump, Mountain, Surveillance, Hammer,
Jar, Teacher, Bicycle, Doctor
```

Among the ambiguous nouns, each noun has two meanings, e.g., *Mouse* can either be an animal or an input device for a computer. This duality in meaning adds a layer of complexity to our dataset, necessitating sophisticated models capable of accurately distinguishing between these varied contexts.

The vocabulary of transitive verbs is:

```
Examine, Emphasize, Catch, Cultivate, Fill, Avoid,
Explore, Admire, Perform, Utilize, Feel, Observe,
Turn on, Uncover, Listen, Eat
```

We used $16+10=26$ qubits in total for each measurement and obtained results for our ambiguity resolution task using both the `qasm_simulator` from `qiskit-aer 0.14.2` and the `ibm_kyoto` quantum computer from IBM. It can be observed in Table III, the left column contains the results from the simulator, while the right column contains the results from `ibm_kyoto`.

Due to the limitation of quantum computing resources, we randomly selected 2 ambiguous nouns and 2 unambiguous nouns for testing on `ibm_kyoto`. In contrast, the `qasm_simulator` was used to test all ambiguous nouns, all of which were perfectly distinguished when preceded by different verbs. This approach highlights the potential of quantum computing for natural language processing tasks, despite current resource constraints.

We observe that on `qasm_simulator`, the accuracy for ambiguity resolution is 100%, while on `ibm_kyoto`, the accuracy falls in the range 82.1% to 98.9% for different verbs. Our results on both simulator and `ibm_kyoto` are in Figure 4

In our setting, we can conclude that the simulator can perfectly disambiguate all the ambiguous words we selected, but a real quantum computer cannot do so perfectly. Although near–term quantum computers are not perfect in understanding human language and processing semantics, we are provided challenges as well as exciting opportunities. By enhancing their capability in understanding human language, we can push the boundaries of current QNLP and unlock the full potential of quantum computers in the future work.

IV. Conclusion

We performed two tasks in QNLP: text classification and ambiguity resolution.

For classification task, we used amplitude encoding and divide–and–conquer algorithms, achieving perfect accuracy on the `lambeq` dataset and 55%–72.5% accuracy on the Amazon review dataset, comparable to current QNLP state–of–the–art results [1]. Using UMAP as dimension reduction on word embeddings yielded the best results in our setting. All classifications were done on the `default.qubit` simulator in `pennylane 0.36.0`.

For ambiguity resolution task, we tested 50 nouns (18 ambiguous and 32 unambiguous) and 18 verbs on the `QASM simulator` in `qiskit 1.1.1`, which perfectly distinguished meanings of ambiguous nouns in context. We also tested 4 random nouns (2 ambiguous and 2 unambigous) on `ibm_kyoto` with 82.1%–98.9% accuracy. Our work extends datasets and improves results of existing ambiguity resolution experiments [1].

Data Availability

All code and data discussed in this work are available in the GitHub repository https://github.com/chizhang24/entangled-meanings. The models used are all freely available online.

Acknowledgment

The authors would like to express sincere gratitude to Larry Moss for his invaluable guidance and Dhananjay Srivastava for helpful discussions, as well as the NLP-Lab (https://nlp-lab.org/) Team at Indiana University Bloomington.

References

[1] D. Widdows, A. Alexander, D. Zhu, C. Zimmerman, and A. Majumder, "Near-Term Advances in Quantum Natural Language Processing," *Annals of Mathematics and Artificial Intelligence*, Apr. 2024.

[2] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997.

dataset	vectorizer	reduction	quantum encoding	num_qubits	accuracy_train	accuracy_val	accuracy_test
lambeq	Word2Vec	PCA	amplitude	4	0.859	0.6154	0.7308
		PCA	divide–and–conquer	4	0.7821	0.8077	0.8846
		TSNE	amplitude	1	0.6282	0.6923	0.5769
		TSNE	divide–and–conquer	1	0.6026	0.6538	0.5385
		UMAP	amplitude	1	0.9359	0.8846	0.9615
		UMAP	divide–and–conquer	1	0.8205	0.9231	0.7308
		NONE	amplitude	8	0.9359	0.9615	0.9615
	spaCy	PCA	amplitude	4	0.6795	0.8462	0.6154
		PCA	divide–and–conquer	4	0.6795	0.7308	0.6923
		TSNE	amplitude	1	0.5897	0.6538	0.6923
		TSNE	divide–and–conquer	1	0.6538	0.6923	0.6923
		UMAP	amplitude	1	1.0	1.0	1.0
		UMAP	divide–and–conquer	1	0.9359	1.0	0.9231
		NONE	amplitude	16	0.9615	1.0	1.0
amazon	Word2Vec	PCA	amplitude	8	0.575	0.625	0.55
		TSNE	amplitude	1	0.475	0.55	0.575
		TSNE	divide–and–conquer	1	0.5417	0.525	0.6
		UMAP	amplitude	1	0.475	0.6	0.575
		UMAP	divide–and–conquer	1	0.5083	0.6	0.725
	spaCy	PCA	amplitude	8	0.5417	0.65	0.6
		TSNE	divide–and–conquer	1	0.475	0.575	0.55
		UMAP	amplitude	1	0.6167	0.625	0.625
		UMAP	divide–and–conquer	1	0.5833	0.75	0.625
		NONE	amplitude	16	0.6167	0.7	0.575

TABLE IV
CLASSIFICATION RESULTS

[3] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, ser. STOC '96. New York, NY, USA: Association for Computing Machinery, Jul. 1996, pp. 212–219.

[4] B. Bauer, S. Bravyi, M. Motta, and G. K.-L. Chan, "Quantum Algorithms for Quantum Chemistry and Quantum Materials Science," *Chemical Reviews*, vol. 120, no. 22, pp. 12 685–12 717, Nov. 2020.

[5] D. Emmanoulopoulos and S. Dimoska, "Quantum Machine Learning in Finance: Time Series Forecasting," Feb. 2022.

[6] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, Sep. 2017.

[7] E. Farhi, J. Goldstone, and S. Gutmann, "A Quantum Approximate Optimization Algorithm," Nov. 2014, arXiv:1411.4028 [quant-ph]. [Online]. Available: http://arxiv.org/abs/1411.4028

[8] C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," *Theoretical Computer Science*, vol. 560, pp. 7–11, Dec. 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304397514004241

[9] D. Peral-García, J. Cruz-Benito, and F. J. García-Peñalvo, "Comparing Natural Language Processing and Quantum Natural Processing approaches in text classification tasks," *Expert Systems with Applications*, vol. 254, p. 124427, Nov. 2024.

[10] B. Coecke, G. de Felice, K. Meichanetzidis, and A. Toumi, "Foundations for Near-Term Quantum Natural Language Processing," Dec. 2020.

[11] P. A. M. Dirac, *The Principles of Quantum Mechanics*, ser. International Series of Monographs on Physics. Oxford, New York: Oxford University Press, Feb. 1982.

[12] D. Jurafsky and J. H. Martin, *Speech and Language Processing (2nd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2009.

[13] J. Lambek, *From Word to Sentence: A Computational Algebraic Approach to Grammar*. Polimetrica s.a.s., 2008.

[14] B. Coecke, M. Sadrzadeh, and S. Clark, "Mathematical Foundations for a Compositional Distributional Model of Meaning," *Lambek Festschirft, special issue of Linguistic Analysis, 2010.*, vol. abs/1003.4394, 2010.

[15] K. Sgarbas, "The Road to Quantum Artificial Intelligence."

[16] "Category Theory in Context." [Online]. Available: https://store.doverpublications.com/products/9780486809038

[17] E. Grefenstette, M. Sadrzadeh, S. Clark, B. Coecke, and S. Pulman, "Concrete sentence spaces for compositional distributional models of meaning," *Proceedings of the 9th International Conference on Computational Semantics (IWCS11)*, pp. 125–134, 2011. [Online]. Available: http://www.aclweb.org/anthology/W/W11/W11-0114.pdf

[18] B. Coecke, B. Edwards, and R. W. Spekkens, "Phase groups and the origin of non-locality for qubits," *Electronic Notes in Theoretical Computer Science*, vol. 270, no. 2, pp. 15–36, 2011, arXiv:1003.5005.

[19] B. Coecke, E. Grefenstette, and M. Sadrzadeh, "Lambek vs. lambek: Functorial vector space semantics and string diagrams for lambek calculus," *Annals of Pure and Applied Logic*, 2013. [Online]. Available: http://arxiv.org/abs/1302.0393

[20] D. Kartsaklis, I. Fan, R. Yeung, A. Pearson, R. Lorenz, A. Toumi, G. de Felice, K. Meichanetzidis, S. Clark, and B. Coecke, "Lambeq: An Efficient High-Level Python Library for Quantum NLP," Oct. 2021.

[21] G. de Felice, A. Toumi, and B. Coecke, "DisCoPy: Monoidal Categories in Python," *Electronic Proceedings in Theoretical Computer Science*, vol. 333, pp. 183–197, Feb. 2021.

[22] R. Lorenz, A. Pearson, K. Meichanetzidis, D. Kartsaklis, and B. Coecke, "QNLP in Practice: Running Compositional Models of Meaning on a Quantum Computer," *Journal of Artificial Intelligence Research*, vol. 76, pp. 1305–1342, Apr. 2023.

[23] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning Word Vectors for Sentiment Analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, D. Lin, Y. Matsumoto, and R. Mihalcea, Eds. Portland, Oregon, USA: Association for Computational Linguistics, Jun. 2011, pp. 142–150.

[24] D. Kotzias, "Sentiment Labelled Sentences," 2015.

[25] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013. [Online]. Available: https://arxiv.org/abs/1301.3781

[26] R. Rehurek and P. Sojka, "Gensim–python framework for vector space modelling," *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, vol. 3, no. 2, 2011.

[27] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, "spaCy: Industrial-strength Natural Language Processing in Python," 2020.

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[29] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction," Sep. 2020.

[30] I. F. Araujo, D. K. Park, F. Petruccione, and A. J. Da Silva, "A divide-and-conquer algorithm for quantum state preparation," *Scientific Reports*, vol. 11, no. 1, p. 6329, Mar. 2021.

[31] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam, G. Alonso-Linaje, B. AkashNarayanan, A. Asadi, J. M. Arrazola, U. Azad, S. Banning, C. Blank, T. R. Bromley, B. A. Cordier, J. Ceroni, A. Delgado, O. Di Matteo, A. Dusko, T. Garg, D. Guala, A. Hayes,

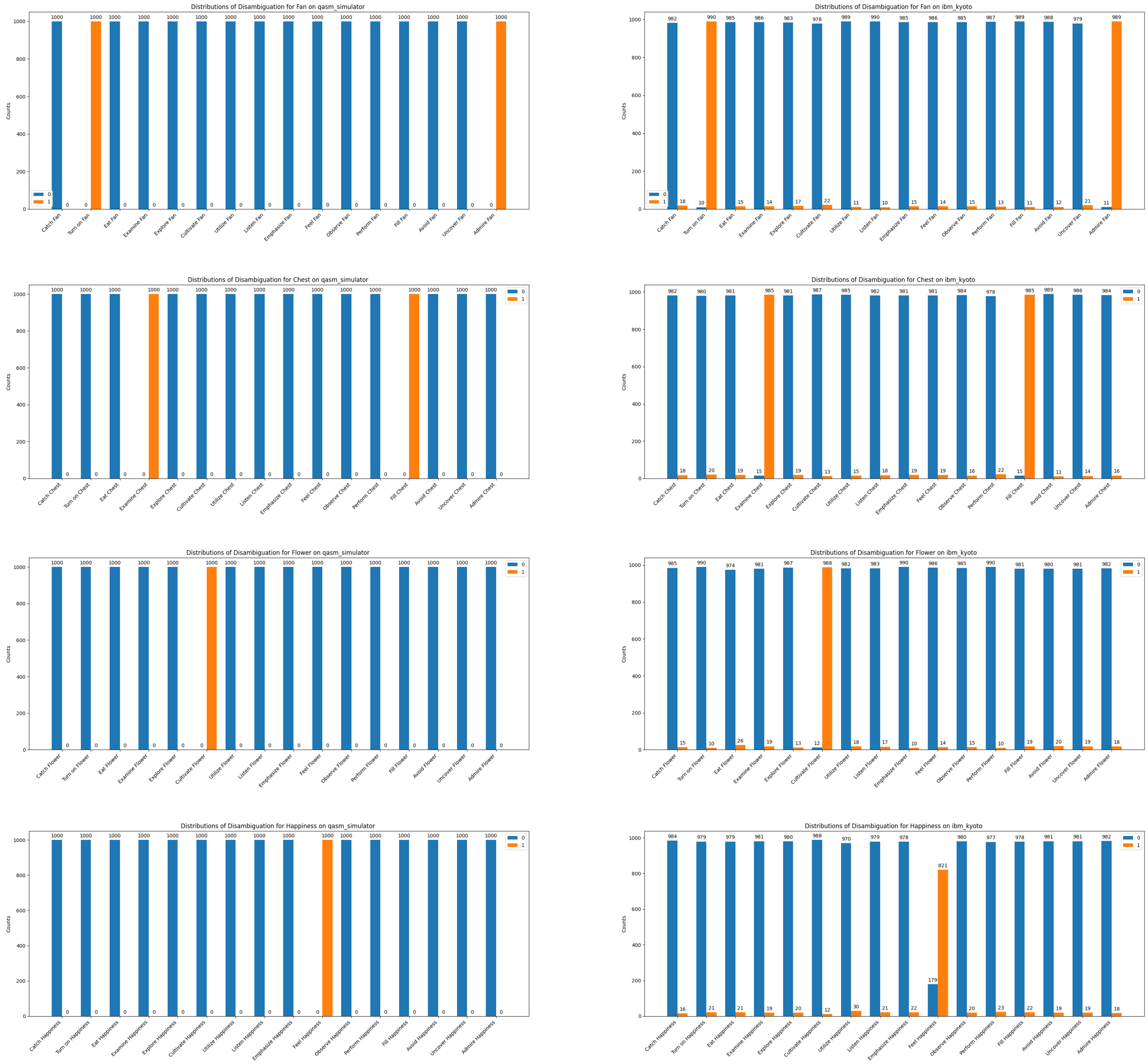

Fig. 4. Results for Ambiguity Resolution on *Fan*, *Chest*, *Flower* and *Happiness*

R. Hill, A. Ijaz, T. Isacsson, D. Ittah, S. Jahangiri, P. Jain, E. Jiang, A. Khandelwal, K. Kottmann, R. A. Lang, C. Lee, T. Loke, A. Lowe, K. McKiernan, J. J. Meyer, J. A. Montañez-Barrera, R. Moyard, Z. Niu, L. J. O'Riordan, S. Oud, A. Panigrahi, C.-Y. Park, D. Polatajko, N. Quesada, C. Roberts, N. Sá, I. Schoch, B. Shi, S. Shu, S. Sim, A. Singh, I. Strandberg, J. Soni, A. Száva, S. Thabet, R. A. Vargas-Hernández, T. Vincent, N. Vitucci, M. Weber, D. Wierichs, R. Wiersema, M. Willmann, V. Wong, S. Zhang, and N. Killoran, "PennyLane: Automatic differentiation of hybrid quantum-classical computations," Jul. 2022.

[32] J. Aitchison, *Words in the Mind: An Introduction to the Mental Lexicon, 3rd Ed*, ser. Words in the Mind: An Introduction to the Mental Lexicon, 3rd Ed. Malden: Blackwell Publishing, 2003.

[33] H. Schütze, "Automatic Word Sense Discrimination," *Computational Linguistics*, vol. 24, no. 1, pp. 97–123, 1998.

[34] M. Baroni and R. Zamparelli, "Nouns are Vectors, Adjectives are Matrices: Representing Adjective-Noun Constructions in Semantic Space," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, H. Li and L. Màrquez, Eds. Cambridge, MA: Association for Computational Linguistics, Oct. 2010, pp. 1183–1193.

[35] Y. Bar-Hillel, *On Categorial and Phrase-structure Grammars*. Weizmann Science Press, 1960.

[36] M. Steedman, *The Syntactic Process*. The MIT Press, Feb. 2000. [Online]. Available: https://direct.mit.edu/books/book/4283/The-Syntactic-Process