Showcasing sentiment classification and word prediction in the quantum natural processing area

David Peral-García [1], Juan Cruz-Benito [2] and Francisco José García-Peñalvo[3]

[1] *Faculty of Science, Universidad de Salamanca (https://ror.org/02f40zc51), Plaza de los Caídos s/n, 37008 Salamanca, Spain*
[2] *IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA*
[3] *GRIAL Research Group, Department of Computers and Automatics, Research Institute for Educational Sciences, Universidad de Salamanca (https://ror.org/02f40zc51), Paseo de Canalejas, 169, Salamanca 37008, Spain*

Abstract

The advent of quantum computers makes it possible to perform quantum computations in different areas like machine learning, finance, or chemistry. This paper showcases one of the emerging areas under quantum machine learning, quantum natural language processing. We present two quantum natural language processing tasks, sentiment classification and missing word prediction in a sentence. We show how these tasks can be achieved even in real quantum computers using the two main libraries in this subfield, DisCoPy, and lambeq.

Keywords

quantum computing, quantum machine learning, quantum natural language processing

1. Introduction and state of the art

Quantum physics and quantum mechanics are nowadays two key components of one of the emerging areas with major theoretical potential in computation: Quantum Computing. As devised in the literature, quantum-computing-based algorithms and procedures like Grover's search algorithm [1] and Shor's algorithm [2] can outperform existing algorithms and solutions and help to scale the existing computational power. Apart from these two typical examples of the potential of quantum computing, some other applications in fields like chemistry, finance, or machine learning have recently received significant attention from the research community. Moreover, some significant results related to these fields exist even with the actual quantum devices, the near-term devices. Quantum machine learning (QML), the process of performing machine learning tasks using quantum mechanics, is one of the emerging areas nowadays. The capabilities of quantum computers could allow machine learning to explore areas that could be too hard to do with classical computing. Nowadays, there exist practical implementations of QML algorithms, like variable depth quantum circuits (vVQC) [3, 4, 5], hybrid quantum autoencoders (HQA) [6], quantum neural networks (QNN) [7, 8, 9] [Figure 1], or hybrid k-neighbours nearby models (HKNN) [10]. A review of the main QML algorithms and its applications between 2017 and 2021 can be found in [11]. A recent subarea of QML is Quantum Natural Language Processing (QNLP), which uses NLP models jointly with certain quantum phenomena such as superposition, entanglement, and interference to perform language-related tasks on quantum hardware. One of the seminal papers about this was written in 2010 by Bob Coecke [12], who established the theoretical and mathematical concepts based on a grammatical theory based on the algebra of Pregroups, introduced by Jim Lambek [13]. The code implementations of this basis are available in DisCoPy and Lambeq libraries [Figure 2].

28th Conference on Information Society and University Studies (IVUS'2023), May 12, 2023, Kaunas, Lithuania
EMAIL: daveral@usal.es (D. Peral-García); juan.cruz.benito@ibm.com (J. Cruz-Benito); fgarcia@usal.es (F. J. García-Peñalvo)
ORCID: 0000-0003-3299-206X (D. Peral-García); 0000-0003-2045-8329 (J. Cruz-Benito); 0000-0001-9987-5584 (F. J. García-Peñalvo)

The first one allows the user to define string diagrams and monoidal functors, and the second one provides different tools to transform and manipulate a sentence, for example, convert it into string diagrams.

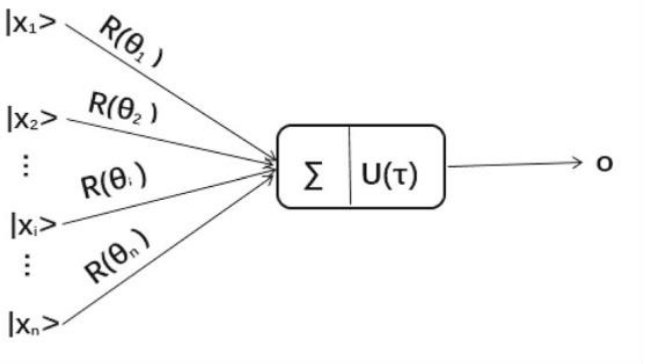


Figure 1: Quantum Neuron Model [7]

Figure 2: Lambeq Pipeline [2]

2. Applications

Using the two libraries cited in the previous section, DisCoPy [14, 15] and Lambeq [16], we are going to present the process to train two different models, one for sentiment classification and one of word prediction.

2.1. Classification

The classification task in machine learning is used to differentiate between two categories or labels; it can be used in images, e.g., classifying between cats and dogs or, as we do in this paper, classifying sentences in positive or negative sentiment. In this section, we start explaining the modeling process and, after that, an example of codification, transform, and training. First, the input sentence is encoded into a string diagram using a parser, for example, DepCGG or BobCat parser [Figure 3]. Second, lambeq allows rewriting the string diagram to reduce complexity, simplify it, and improve performance in the future training step. Third, the diagram is parameterized, and we convert it into a circuit using an ansatz. Some examples of the ansatzs that lambeq provides are SpiderAnsatz [Figure 4], IQPAnsatz [Figure 5], Sim14Ansatz, Sim15Ansatz or StronglyEntanglementAnsatz. Fourth, when the parameterisable circuit is created, a compatible backend with the model must be defined. For example, we can use a quantum backend like the qiskit [17] backend with the TketModel, or we can use classical resources (compatible with Jax [18] and GPU) with the PytorchModel. Finally, the trainer has different options to personalize and adapt to the model like a typical classical trainer: loss functions, epochs, optimizers, hyperparameters, and evaluation functions.

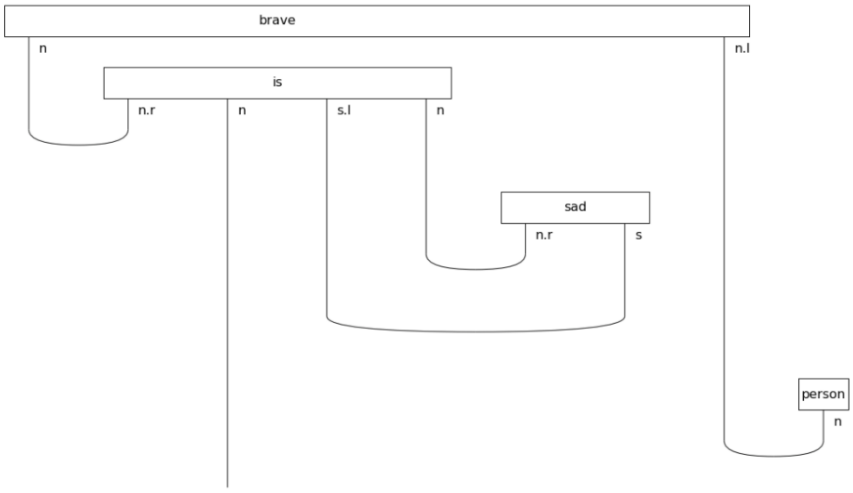


Figure 3: String Diagram

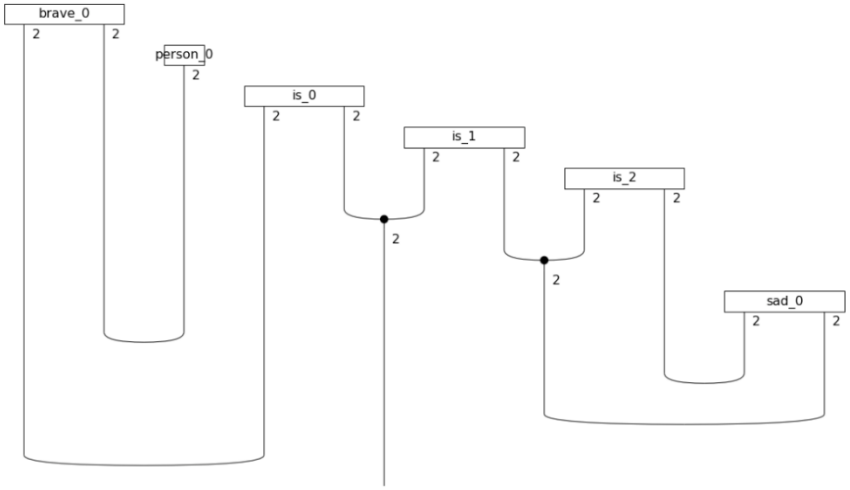


Figure 4: Spider Ansatz

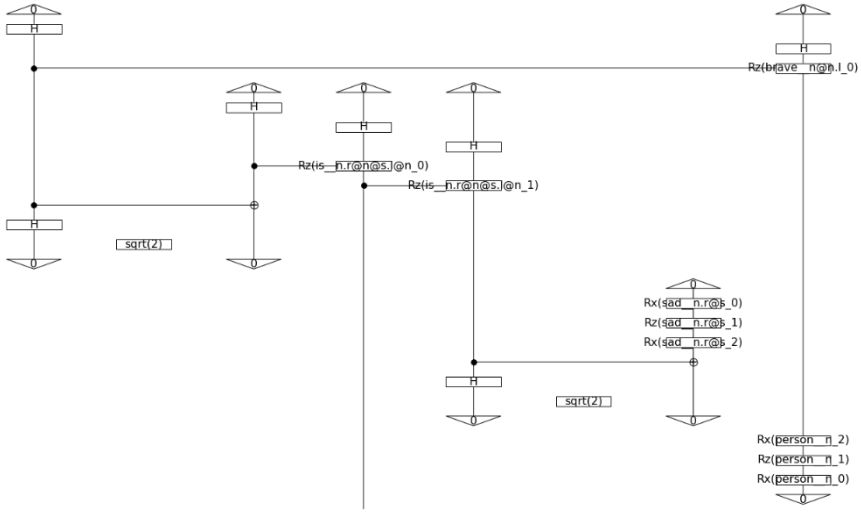


Figure 5: IQP Ansatz

2.2. Prediction

The prediction task is used to obtain an unknown value based on the previous historical data. A model is trained with prior data and tries to predict the missing data with a certain probability percentage. One of the areas that widely use this technology is business intelligence which tries to predict the future behavior of the market and make decisions accordingly. In this case, we try to predict the missing word from a sentence based on previous data we input into the model. With the DisCoPy library, we tried to predict one of the words that can follow up the sentence based on the input. First, we select a sentence with a defined grammatical structure. In this case, we defined the structure 'noun' + 'transitive verb' + 'noun', e.g., 'Alice drinks water' [Figure 6]. Second, we removed the word of the sentence we want to predict and replaced it with an unknown value, e.g.,'?'. 'Alice drinks '?' [Figure 7]. Third, we trained the model with this unknown value and a corpus of words that could be the correct answer. When the model is trained, we check if the results are correct [Figure 8].

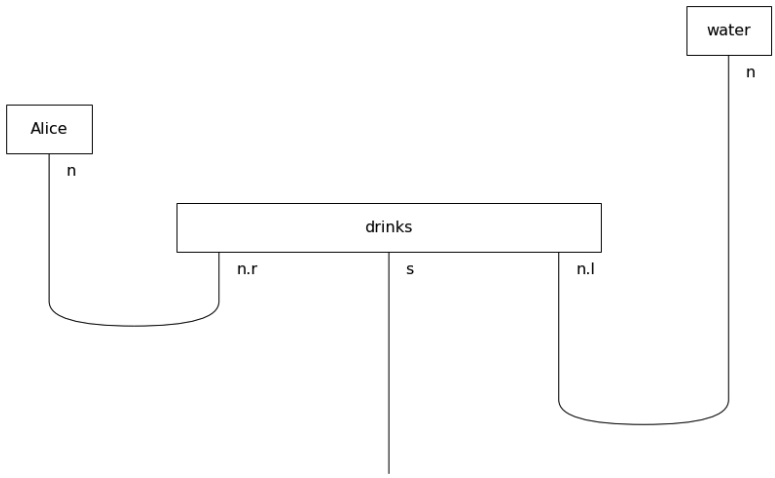


Figure 6: Prediction Model Input Sentence

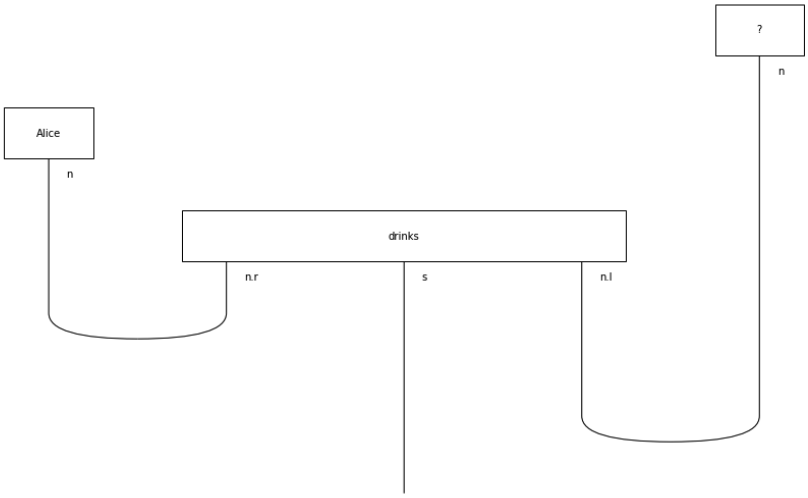
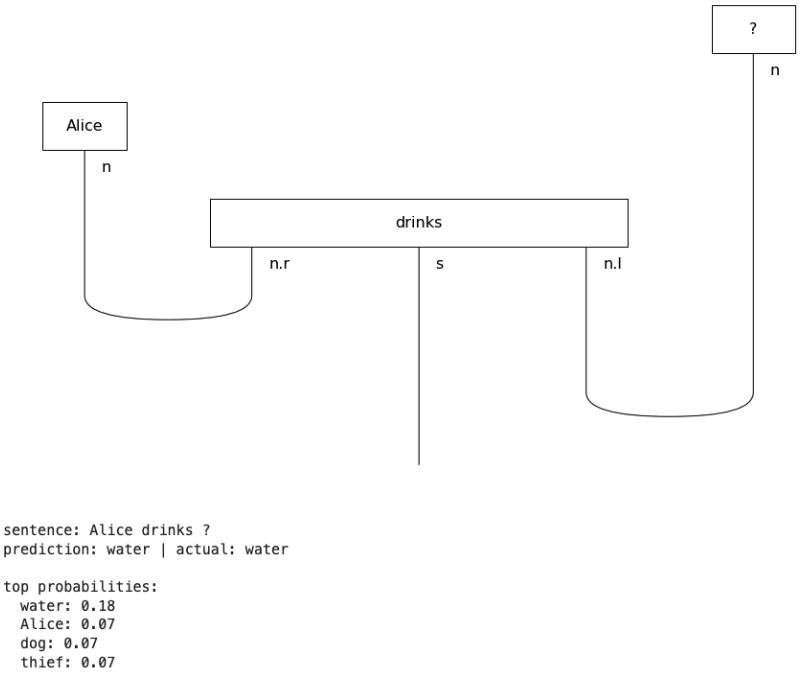


Figure 7: Prediction Model Missing Value




```
sentence: Alice drinks ?
prediction: water | actual: water

top probabilities:
  water: 0.18
  Alice: 0.07
  dog: 0.07
  thief: 0.07
```

Figure 8: Prediction Model Results

3. Conclusions and Future Implementations

As we cited in the introduction and state-of-the-art section, the conceptual and mathematical basis of QNLP had been established in the literature. Based on them, we presented some proof of concepts for QNLP classification and prediction models with libraries that implement this basis: DisCoPy and lambeq. In this paper, we demonstrate that it is possible to compute classification and prediction tasks using quantum computing. Even if the tasks are grammatically simple and this type of task had already been resolved by classical computing, there is a solid foundation that will enable us to increase the complexity of the tasks in future experiments. On the one hand, regarding the classification task, we use a parser to transform the sentences into string diagrams, allowing us to analyze each sentence's grammatical structure. Once this step is complete, we transform this string diagram into an ansatz; this step is required to parameterize the object to train a model. Finally, depending on the backend we are using to train the model; we select an ansatz compatible with classical or quantum devices.

On the other hand, the prediction task has the same first step as classification, converting the input data, the sentence, into a string diagram that represents the grammatical structure. In this proof of concept, we remove one word from the sentence and try to predict it. We can observe that the model predicts the correct word but with a low probability (18%). In future implementations of the examples presented, for classification tasks, we should increase the complexity of the sentences we use in the input data of the models or try to classify more than one sentence by joining them using, for example, DisCoCirc [19]. In the prediction field, our primary goal is to increase the accuracy probability of this type of task in future experiments to achieve a model comparable to classical models. Also, we can try to predict more than one word for each sentence or without the grammatical context of the predicted word. The main challenge in this area is to compute and train a model with a complete and larger text, which is not already viable due to the number of qubits available in current quantum devices. One of the potential strategies to address this problem, in parallel to the evolution of quantum hardware, is the development of methods that allow us to encode the grammatical and semantic meaning of the sentence using the fewer qubits possible, generating new QNLP procedures and algorithms [20] or following the path of research advances in other quantum computing areas [21].

4. References

[1] L. K. Grover, A fast quantum mechanical algorithm for database search, in: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC '96, Association for Computing Machinery, New York, NY, USA, 1996, p. 212–219. URL: https: //doi.org/10.1145/237814.237866. doi:10.1145/237814.237866.

[2] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Journal on Computing 26 (1997) 1484–1509. URL: http: //dx.doi.org/10.1137/S0097539795293172. doi:10.1137/s0097539795293172.

[3] Y. Huang, H. Lei, X. Li, Q. Zhu, W. Ren, X. Liu, Quantum generative model with variable depth circuit, Computers, Materials and Continua 65 (2020) 445–458. doi:10.32604/cmc. 2020.010390.

[4] M. Benedetti, D. Garcia-Pintos, O. Perdomo, V. Leyton-Ortega, Y. Nam, A. Perdomo-Ortiz, A generative modeling approach for benchmarking and training shallow quantum circuits, npj Quantum Information 5 (2019). URL: http://dx.doi.org/10.1038/s41534-019-0157-8. doi:10.1038/s41534- 019- 0157- 8.

[5] M. Schuld, A. Bocharov, K. M. Svore, N. Wiebe, Circuit-centric quantum classifiers, Physical Review A 101 (2020). URL: http://dx.doi.org/10.1103/PhysRevA.101.032308. doi:10.1103/ physreva.101.032308.

[6] M. Srikumar, C. D. Hill, L. C. L. Hollenberg, Clustering and enhanced classification using a hybrid quantum autoencoder, Quantum Science and Technology 7 (2021) 015020. URL: https://dx.doi.org/10.1088/2058-9565/ac3c53. doi:10.1088/2058- 9565/ac3c53.

[7] B. Q. Chen, X. F. Niu, Quantum Neural Network with Improved Quantum Learning Algorithm, International Journal of Theoretical Physics 59 (2020) 1978–1991. doi:10.1007/ s10773- 020-04470- 9.

[8] F. Tacchino, S. Mangini, P. K. Barkoutsos, C. Macchiavello, D. Gerace, I. Tavernelli, D. Bajoni, Variational learning for quantum artificial neural networks, IEEE Transactions on Quantum Engineering 2 (2021) 1–10. URL: http://dx.doi.org/10.1109/TQE.2021.3062494. doi:10.1109/ tqe.2021.3062494.

[9] J. Bausch, Classifying data using near-term quantum devices, International Journal of Quantum Information 16 (2018). doi:10.1142/S0219749918400014.

[10] M. L. LaBorde, A. C. Rogers, J. P. Dowling, Finding broken gates in quantum circuits: exploiting hybrid machine learning, Quantum Information Processing 19 (2020). doi:10. 1007/s11128- 020-02729- y. arXiv:2001.10939.

[11] D. Peral-García, J. Cruz-Benito, F. J. García-Peñalvo, Systematic literature review: Quantum machine learning and its applications, 2022. arXiv:2201.04093.

[12] B. Coecke, M. Sadrzadeh, S. Clark, Mathematical foundations for a compositional distributional model of meaning (2010). URL: https://arxiv.org/abs/1003.4394. doi:10.48550/ ARXIV.1003.4394.

[13] J. Lambek, The mathematics of sentence structure, The American Mathematical Monthly 65 (1958) 154–170. URL: https://doi.org/ 10.1080/00029890.1958.11989160. doi:10.1080/00029890.1958.11989160. arXiv:https://doi.org/10.1080/00029890.1958.11989160.

[14] G. de Felice, A. Toumi, B. Coecke, DisCoPy: Monoidal categories in python, Electronic Proceedings in Theoretical Computer Science 333 (2021) 183–197. URL: https://doi.org/10.4204%2Feptcs.333.13. doi:10.4204/eptcs.333.13.

[15] A. Toumi, G. de Felice, R. Yeung, Discopy for the quantum computer scientist, 2022. URL: https://arxiv.org/abs/2205.05190. doi:10.48550/ARXIV.2205.05190.

[16] D. Kartsaklis, I. Fan, R. Yeung, A. Pearson, R. Lorenz, A. Toumi, G. de Felice, K. Meichanetzidis, S. Clark, B. Coecke, lambeq: An Efficient High-Level Python Library for Quantum NLP, arXiv preprint arXiv:2110.04236 (2021).

[17] Qiskit contributors, Qiskit: An open-source framework for quantum computing, 2023. doi:10.5281/zenodo.2573505.

[18] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, Q. Zhang, JAX: composable transformations of Python+NumPy programs, 2018. URL: http://github.com/google/jax.

[19] B. Coecke, The mathematics of text structure, 2020. arXiv:1904.03478.
[20] V. Wang-Mascianica, J. Liu, B. Coecke, Distilling text into circuits, 2023. arXiv:2301.10595.
[21] A. Eddins, M. Motta, T. P. Gujarati, S. Bravyi, A. Mezzacapo, C. Hadfield, S. Sheldon, Doubling the size of quantum simulators by entanglement forging, PRX Quantum 3 (2022) 010309.