Near-Term Advances in Quantum Natural Language Processing

Dominic Widdows, Daiwei Zhu, Chase Zimmerman IonQ, Inc.

June 7, 2022

Abstract

This paper describes experiments showing that some problems in natural language processing can already be addressed using quantum computers. The examples presented here include topic classification using both a quantum support vector machine and a bag-of-words approach, bigram modeling that can be applied to sequences of words and formal concepts, and ambiguity resolution in verb-noun composition.

While the datasets used are still small, the systems described have been run on physical quantum computers. These implementations and their results are described along with the algorithms and mathematical approaches used.

1 Introduction

Natural Language Processing (NLP) is a promising area of application using quantum computers. This paper reports on progress in developing and running NLP experiments on trapped-ion quantum computers during the first few months of 2022. The experimental applications include topic classification, bigram modeling, and ambiguity resolution.

More generally, building real artificial general intelligence is a core goal of computer science. Reasons for believing that quantum models and quantum computing may contribute to building more fully intelligent systems are varied, and are discussed in Section 2. Our approach to building early-stage language processing systems on quantum computers is motivated by these goals. Questions this entails include: what properties are lacking in artificial intelligence (AI) today; whether these gaps can be addressed with quantum approaches; what core mathematical / language operations can be identified; and for each operation, what are the simplest effective implementations? These questions help to motivate medium-term challenges and immediate progress.

The examples presented in this paper include word-based topic classification in Section 3, bigram sequence modelling in Section 4, and ambiguity resolution in verb-noun composition in Section 5. Progress in related systems and fields is surveyed briefly in Section 6. The datasets and systems so far are small, but the results are accurate and a motivation for much further work. Along with the results on semantic composition presented by Lorenz et al. (2021), this sets the stage for rapid progress in quantum NLP within the next decade.

2 The Promise of Quantum NLP

Overlaps between quantum theory and natural language processing are many and varied, ranging from structural and qualitative similarities to directly sharing mathematical techniques.

2.1 Structural Similarities

Language and quantum mechanics have notable qualitative similarities (Widdows, 2003a). Most words have several possible meanings or could refer to several different things. Even if we can predict the range of options in advance, we do not typically know which interpretation of a word or phrase is appropriate until we encounter it in context. However, once a word is observed in context, one of the available meanings is typically selected, and this selection tends to remain fixed until the context changes. Similarly, quantum mechanics predicts the probabilities of different outcomes, not which outcome is actually found, unless the system is already in a fixed state that is aligned with the measurement being performed. When measured, the system is observed to be in a fixed state, and if measured again, the same result will be given — the outcome remains fixed until something changes it (Dirac, 1930, Ch 2). Even if we prefer precise deterministic theories in physics and linguistics, nature and language are unpredictable in practice.

Features found in quantum theory have been explored in other human sciences, noting similar patterns. The nature of money in economics and finance is one example: Orrell (2020) argues that credits and debts show similarities to the concept of quantum entanglement, and that money, price, and value are uncertain properties that only become fixed when they are agreed and recorded in transactions, likening the former to quantum and the latter to classical information. Cognitive scientists have measured relationships between available information, context, and decision-making, that violate classical probabilistic laws but can be represented concisely and accurately as phase interference terms in quantum models (Busemeyer and Bruza, 2012).

Such similarities do not show us how to perform particular language processing tasks, but they do suggest that there is a rich space to explore, which is especially inviting for problems involving large amounts of context and uncertainty, which have sometimes frustrated classical or more reductionist approaches. For some human problems, if we look to mathematics and physics for helpful techniques at all, quantum mechanics and logic are sometimes at least as intuitive a place to start as classical mechanics and logic.

2.2 Compositional Behaviors

Language is compositional, in the sense that the meanings of new phrases and sentences can be derived from their parts, even when the phrase itself has not been seen before or describes something quite unrealistic. For example, one may have never heard the phrase "purple polar bear" before, but it immediately conjures up a mental picture of a polar bear whose color is purple.

Language composition sometimes produces similar structures from dissimilar ingredients — for example, in the context of hotels, "book a room" and "make a reservation" have overlapping and sometimes identical meanings, even though neither of the pairs (book, make) or (room, reservation) are usually synonyms. The phrases nonetheless make immediate sense to familiar listeners, at which point it becomes harder to decompose the phrase into its constituents. Perhaps the word "book" is there because there used to be physical books that such plans was written down in, and at that point the reservation was "booked". The claim here is not that this is the true etymology, but that the exercise of looking for such a piecewise explanation

is itself unusual for us.

Schrödinger (1935) famously noted a similar phenomenon in quantum mechanics:

When two systems, of which we know the states by their respective representatives, enter into temporary physical interaction due to known forces between them, and when after a time of mutual influence the systems separate again, then they can no longer be described in the same way as before, viz. by endowing each of them with a representative of its own. I would not call that one but rather the characteristic trait of quantum mechanics, the one that enforces its entire departure from classical lines of thought.

Similar patterns have been noted for centuries — for example, Aristotle (*De Interpretatione*, Ch 2) argues that in the common phrase "a fair steed", the word "steed" has an independent meaning, but in the proper name "Fairsteed", it does not. ¹ Given the prevalence of composition as a general topic, one might ask whether any overlap between language and quantum mechanics here is just accidental. One mathematical approach to answering this question has been to explore common compositional structures using category theory. This has become a fertile direction for research, because common structures have been identified (in particular, compact closed categories), and moreover, this has shown ways in which various grammatical and lexical structures can be represented and even combined into tensor networks that can be implemented as quantum circuits (Piedeleu et al., 2015; Coecke et al., 2020).

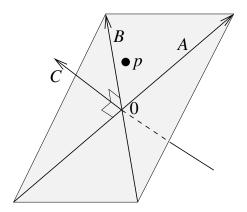
2.3 Vectors Everywhere

Vectors and linear algebra have become ubiquitous throughout artificial intelligence, including NLP, important cases including the use of feature vectors in statistical machine learning, and the representation of states and operators in neural networks (Géron, 2019). Linear algebra is also central to quantum mechanics and quantum computing (Dirac, 1930; Nielsen and Chuang, 2002, Ch 2).

Again one may ask whether these overlaps are significant — the use of vectors alone might be a shallow commonality. But many more pieces of quantum mathenmatics have been found to useful in language-related tasks, including subspaces and projections for representing conditionals (Van Rijsbergen, 2004) and logical negation and disjunction (Widdows, 2004), quantum probability and the analysis of off-diagonal correlations for information retrieval (Sordoni et al., 2013), entanglement for representing the combined representation of many pairwise relationships (Cohen et al., 2012).

While vector spaces have become ubiquitous in AI, some of the opportunities they present have remained comparatively unused. For example, the lattice of subspaces (lines, planes, and higher-dimensional versions of these) provides a geometric model for a kind of logic, just as set theory gives a model for Boolean logic (Figure 1). This was discovered by Birkhoff and von Neumann (1936) and given the name "quantum logic". In terms of operator algebras, the subspaces are in one-to-one correspondence with the lattice of *projectors* onto those subspaces, and projections move points around the space. This gives quantum logic a *non-local* character, which can be exploited for semantic modelling. The negation operator in quantum logic is projection onto the orthogonal subspace — for example, in Figure 1, "p NOT (A OR B)" would be represented as the orthogonal projection of the point p onto the line C, since C is orthogonal to the plane spanned by A and B. This form of vector negation can be used to navigate between different areas of meaning, rather than just removing unwanted items piecemeal (Widdows, 2004, Ch 7). Several techniques

 $^{^{1}}$ The words Aristotle uses are 'Κάλιππσς' contrasted with 'χαλός ἵππος', χαλός as in 'calligraphy' and ἵππος as in 'hippodrome' or 'hippopotamus', so the correspondence with English "fair steed" is quite direct (Cooke and Treddenick, 1938, p. 116).



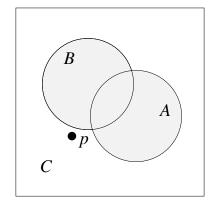


Figure 1: Comparing Quantum with Boolean Logic

that have been adapted from earlier quantum literature and applied to problems in AI today. Many of the successes of quantum mathematics in artificial intelligence to date are surveyed by Widdows et al. (2021).

2.4 The Availability of Quantum Computers

Far from being considered popular or fashionable, research using quantum mathematical models outside of conventional platforms of quantum physical systems such as isolated atoms have often been met with skepticism and resistance — perhaps because quantum theory *is* too-often overhyped and confused in general science writing, and in some cases because clinging to classical determinism is considered to be much more scientifically respectable (Orrell, 2020, Ch 1).

These hurdles may be becoming easier to surmount — for example, in the past year, major journals have published papers on quantum approaches in areas including at least economics (Orrell and Houshmand, 2022), psychology (Pothos and Busemeyer, 2022), and AI (Widdows et al., 2021). Reasons for this acceptance must include the reality of quantum hardware. Working quantum computers have been built and are running programs every day — in a development process spanning decades, concepts like "entangled pairs" have gone from scientific imagination through experimental demonstration to engineering implementation. We may be uncomfortable with such quantum concepts, but unlike pioneers in the 1930's, we have evidence that resolves any reasonable dispute about their reality.

The question for quantum approaches is no longer "Are they real?" but "Are they useful?". In NLP the work is still in its early days, and success must be demonstrated by building useful systems. Rather than predicting a singular quantum advantage for a particular QNLP system, this paper explores several examples, with an eye to general capabilities. The next few sections give examples of how standard language processing tasks have been successfully performed on quantum computers. Experiments below used the 11-qubit trapped ion quantum computer described by Johri et al. (2021), and where necessary the larger IonQ Aria machine with a capacity of 32 physical and 20 algorithmic qubits (IonQ Aria, 2022).

3 Word-Based Topic Classification and State Addition

Text classification or topic classification is the task of assigning topic labels to texts such as SPORTS or MUSIC, or in the case of "Which bands performed at the Super Bowl this year?", perhaps both. It has many uses including intent classification in dialog systems, assigning topic keywords to documents, routing incoming

messages for customer service, and spam filtering. There are many successful methods, often involving vector representations and various forms of neural networks and dimensionality reduction (Kowsari et al., 2019).

3.1 One Qubit Per Word and Topic

The quantum implementation developed here was deliberately simple: a bag-of-words classifier on a quantum computer. In this context, 'bag-of-words' means that only individual words are used as features for the classifier: the order of words is not taken into account, or even which combinations of words appear. So for example, a bag-of-words classifier may predict that a document containing the word *horse* is about FARMING or SPORTS, which will often be correct, but is a mistake if it occurs in the phrase "A horse chestnut is a deciduous tree with palmate leaves."

The training process for such a classifier is usually some form of supervised machine learning. A collection of texts or documents is sampled, labels are applied to these texts by human annotators, and these examples are used to train the classifier to deal with new unseen examples. For a bag-of-words classifier, a simple version of this process is:

- 1. For each (document, topic) pair in the training set, split the document text into a list of words.
- 2. For each word in that list, increment a score recording how often this word was seen with this topic.

This process gradually builds up a term-topic matrix (if each document is considered to be a distinct topic, this is exactly the same as a term-document matrix). The classification stage then proceeds as follows:

- 1. A new document is passed to the classifier. It is split into a list of words.
- 2. For each word, look up the scores that word acquired for each topic during training.
- 3. Given each topic an overall score accumulated from the sum of the scores for each word.
- 4. Choose the topic with the highest score as the overall winner.

This example was chosen because it raises some basic mathematical implementation questions — in particular, item 2 in training, "How do we keep a score for each (word, topic) pair?" and item 3 in classification, "How do we accumulate an overall score from these individual contributions?" The first of these problems can be addressed using repeated single-qubit rotations, and the second by connecting the rotated qubits to a common 'sum qubit', as shown in Figure 2. A detailed analysis of the gate matrices in Figure 2 shows that the probability of measuring a state $|1\rangle$ in the sum qubit q_2 is given by $\sin^2(\theta)\cos^2(\varphi)+\cos^2(\theta)\sin^2(\varphi)$. This is the case when the input rotations are X-rotations (Nielsen and Chuang, 2002, §1.3.1). Using other fractional rotations as generators gives combinations with different algebraic properties, investigated more thoroughly in a mathematical paper (Widdows, 2022). Most generally, if the two single-qubit rotations in Figure 2 are represented by the matrices

$$A = \begin{bmatrix} \alpha & \beta \\ -\overline{\beta} & \overline{\alpha} \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} \gamma & \delta \\ -\overline{\delta} & \overline{\gamma} \end{bmatrix},$$

then

$$P(q_2 = |1\rangle) = |\alpha \overline{\delta}|^2 + |\overline{\beta}\gamma|^2.$$

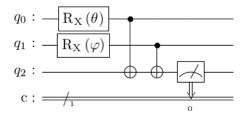


Figure 2: Example Adder Circuit that Combines the Angles θ and φ

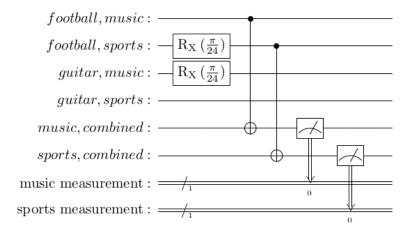


Figure 3: Example Classification Circuit for Two Words and Two Topics

The bag-of-words classifier is built by assembling several such qubits and gates. During training, each (word, topic) pair is assigned to a particular qubit, and the (word, topic) weights are accumulated by scanning over the training set and incrementing the rotation for the corresponding qubit every time a given word occurs with a given topic. For each topic, an extra qubit is declared to keep the sum for each topic, so the number of qubits required is $(\text{num_words} + 1) \times \text{num_topics}$. (Hence this design only works as a demonstration for very small vocabularies.) During classification of a new phrase, each recognized word in the phrase has each of its topic weights connected to the sum qubit for that topic.

As a worked example, consider the toy training corpus:

I play football \rightarrow SPORTS I play guitar \rightarrow MUSIC

The training process records that the content word *football* gives a score to SPORTS and *guitar* gives a score to MUSIC (assuming a preprocessing step that notes that the other words are common to all topics.) When classifying the phrase "Do you want to play football?", the word *football* is recognized and each of the *football, topic* weights are connected to the sum qubit for the corresponding topic. Each of the topic qubits is measured, and the winner is the topic that measures the most $|1\rangle$ states over a number of shots. A circuit implementing this process is shown in Figure 3.

3.2 One Qubit Per Embedding Dimension

The design above is clear but wasteful — a system that requires distinct bits for each word in the vocabulary would be fine in classical but not yet in quantum computing. In machine learning terms, using a qubit for each word is an example of a 'one-hot encoding', and to avoid redundancy and get more information out

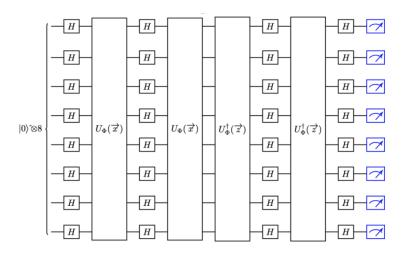


Figure 4: SVM Circuit Example with 8-Dimensional Input

of each dimension, more compact distributional vector embeddings are often preferred. In this experiment, the Word2Vec technique was used (Mikolov et al., 2013) to reduce the representation of each word to 8 dimensions. This part of the process was run classically as a preprocessing step.

These vectors were then used as inputs for a feature map that was then used as input for a quantum support vector machine. For these, the <code>ZZFeatureMap</code> and <code>QSVM</code> implementations from the Qiskit package were used (ANIS et al., 2021), as shown in Figure 4. As observed in the results below, the classical embeddings preprocessing step enabled the quantum circuit to achieve relatively high accuracy with fewer qubits.

3.3 Data and Experimental Results

Due to their limited scale, these systems could only be evaluated on very small datasets. For this, the test and training datasets from the lambeq package were used (Kartsaklis et al., 2021), which was used for the evaluation of the compositional quantum NLP system of Lorenz et al. (2021). The dataset consists of 70 training sentences and 30 dev and test sentences, artificially generated to use a small fixed vocabulary and to follow predictable syntactic patterns. Lorenz et al. (2021) report an accuracy of 83.3% on the test dataset using a 6-qubit quantum computer, and Kartsaklis et al. (2021) report achieving 100% accuracy after several rounds of training on a noisy simulation of a quantum computer. These are the nearest comparable experiments to those reported here, so the same dataset was deliberately reused.

Both the "one qubit per word and topic classifier" of Section 3.1 was able to reach 100% accuracy, and the word embeddings classifier of Section 3.2 was able to reach 90% accuracy, with comparatively little training and tuning.

For the bag-of-words classifier of Section 3.1, the key parameters to tune were the number of words to use and the incremental rotation angle. The tradeoff here is that too small an incremental rotation can be insignificant compared to statistical noise, and too large an incremental rotation can lead to cumulative rotations that go too far and 'turn the corner' — a naive version of classical 'buffer overflow' in a single qubit. The incremental rotation angle was chosen to be $\pi/24$ (as in Figure 3) because this gave accurate results on the development set. Larger datasets would obviously warrant smaller angles to avoid the overflow issues, and this parameter tuning could easily be automated for a larger variational learning approach. The

number of words used is partly a tradeoff between space and accuracy. The largest system available to us had 20 algorithm qubits (IonQ Aria, 2022), and offline simulations showed that perfectly accurate results could be expected with at least 9 words. To build a circuit like that of Figure 3 with 9 words and 2 topics requires 20 qubits, so by a convenient accident, this was just able to run comfortably on available quantum hardware, yielding 100% accuracy.

For the word embeddings classifier of Section 3.2, only one qubit per dimension was needed, and results were experimentally found to be 90% accurate with just 8 embedding dimensions, thus requiring only 8 qubits to run.

3.4 Language Caveats and Dependencies

As well as the simple data size and scaling questions, there are many other language details to be considered, including "What does splitting into words mean precisely?" (tokenization) and "How do we declare that two words are the same?" (canonicalization). These questions were sidestepped for now by using particularly simple English phrases as the training and test data. However, it is still important to bear in mind that all the approaches require classical preprocessing of various sorts, including basic whitespace-splitting and assigning tokens to qubit indices in the word-based classifier, word vector embedding training for the inputs to the SVM classifier, to the need for a full natural language parse tree in the lambeq package.

Rather than claiming that the bags-of-words classifier is 'the best' because it reaches the highest accuracy on a quantum hardware, we believe these comparisons highlight important tradeoffs to consider. With a much more sophisticated linguistic model and related requirements for preprocessing, Lorenz et al. (2021) were able to reach 87% accuracy with only 6 qubits — whereas with the same restriction (implemented by reducing the number of words considered), the bag-of-words classifier accuracy was reduced to 83% in simulations. Having a range of available approaches puts us in a position where tradeoffs between space requirements, system complexity, and results accuracy can be considered in the light of what problems users are trying to solve.

4 Bigram Modelling and Quantum Probability

While the ability to combine classifier weights and get good results is important, it is obvious and well-known that bag-of-words models are a poor approximation for language. Historically, one of the next steps in statistical language modelling was the modelling language based on short sequences of characters and words, used by Shannon (1948, §3) to illustrate key behaviors including entropy in the foundation of information theory. Shannon used the terms 'first-order approximation' for a model that samples and generates words based purely on their frequencies, and 'second-order approximation' for a model that conditions this probability on the preceding word. (For example, *yellow* is likely to come before but not after *submarine*.) Today these are often called unigram and bigram models, and in principle such a model extends to sequences of any length, leading to the long-established field of n-gram language modelling (Manning and Schütze, 1999, Ch 6). Having built a quantum unigram model, extending this to bigram modelling is a natural next step.

Though still relatively simple, semantic modelling for two-word bigram combinations has real-world applications. For example, in the hospitality domain, *book room*, *arrange accommodation*, *reserve hotel*, *make reservation* all imply the same intent or goal, even though they do not share the same words and these

words are often not synonyms in other contexts — for example, *book* and *make* are not normally synonyms in English. Recognizing that these phrases mean similar things when combined in this way is this important for intent recognition in customer service dialog systems.

In probabilistic terms, a unigram model can be used for language generation by sampling terms from the distribution of words, in which case a bigram model works by sampling from the *joint* distribution of (ordered) word pairs. This has an established parallel in quantum probability theory, described in detail by Bradley (2020). A joint distribution over the vector spaces V and W is represented in the tensor product space $V \otimes W$, in which case the marginal distributions are represented by partial traces defined by $\operatorname{tr}_W(v \otimes w) = v \operatorname{tr}(w)$ and $\operatorname{tr}_V(v \otimes w) = w \operatorname{tr}(v)$. Bradley (2020) uses this approach to represent formal concept lattices in vector spaces. Formal concept analysis is a theory developed by Ganter and Wille (1999) in which objects are represented by which attributes they possess, and concepts arise as closures of sets of objects and attributes that go together.

4.1 Modelling Steps

The distributional modelling process outlined by Bradley (2020) uses the following steps, which are written out fully in the example in Table 1.

- 1. Write down the joint distribution matrix ('Initial Distribution'). In the example in Table 1, this joint distribution is a subset of the Cartesian product Colors × Fruit.
- 2. Express this distribution as an element of the tensor product space Colors ⊗ Fruit by flattening the distribution matrix into a long vector.
- 3. Normalize this element to get a distribution state vector $|\psi\rangle$.
- 4. Take the outer product $|\psi\rangle\langle\psi|$ which gives the density matrix of the joint distribution.
- 5. Take partial traces of this density matrix. Each partial trace reveals information about the distribution of the respective ingredients in particular, the diagonal elements represent the individual marginal probabilities, and the off-diagonal elements represent information about correlations between items.

In the worked example of Table 1, the key thing to note is the off-diagonal $\frac{1}{3}$ terms that link the *red* and *green* concepts in the partial trace showing colors. This reflects the fact that back in the original distribution, these are both colors of *apple*. Note that classical marginalization leaves such information behind — the classical marginal distribution would just tell us that *red*, *green*, and *yellow* are equally likely. This property has been noted by various authors, and was put to work in the work of Sordoni et al. (2013) for term-weighting in information retrieval. Those authors recognized that the weight given for a combination of words should sometimes be different from the additive weights of the individual words — taking an example from earlier, the weight of the term *horse* in the phrase "I love the flowers on horse chestnut trees" should be suppressed because it is unlikely to be relevant for users whose search terms include *horse* on its own.

4.2 Quantum Circuit Approximation

The initial quantum implementation goal was to learn to fit a joint bigram distribution in a way that can potentially be used for language understanding and generation. For this, a slightly larger distribution of

Table 1: Steps for modelling the concepts "red apple", "green apple", "yellow banana"

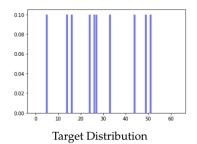
$$\operatorname{tr}_{\mathsf{Fruit}}(|\psi\rangle\,\langle\psi|) = \begin{array}{c|c} \operatorname{red} & \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 \\ 0 & 0 & \frac{1}{2} \end{array} \right] \qquad \operatorname{tr}_{\mathsf{Colors}}(|\psi\rangle\,\langle\psi|) = \begin{array}{c|c} \operatorname{apple} & \frac{2}{3} & 0 \\ \operatorname{banana} & \begin{bmatrix} \frac{2}{3} & 0 \\ 0 & \frac{1}{3} & \frac{1}{3} \end{array} \right]$$

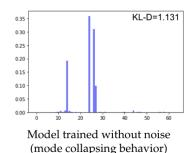
items and colors was used including *green apple*, *red apple*, *yellow banana*, *ripe banana*, *red pepper*, *yellow pepper*, *black shoes*, *red dress*, *blue suit*, *white shirt*.

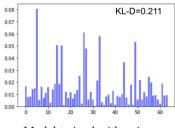
We use a Quantum Circuit Born Machine (QCBM) model to learn the joint probability distribution. The QCBM encodes the probability of (prefix,suffix) pairs directly as the amplitudes of different states in the superposition. This way, the normalization is satisfied by default. For a joint distribution with n prefixes and m suffixes, we would need nm classical bits just to model 'present or absent'. For the QCBM model, we use $\lceil \log_2(n) \rceil + \lceil \log_2(m) \rceil$ qubits, and an encoding convention where each prefix and suffix is given a binary index, and the cooccurrence weight is stored as the amplitude of the state given by concatenating these two indices. Each of the adjectives and nouns was assigned a binary index corresponding to a tensor product state, for example $blue = |101\rangle$ and $shoes = |100\rangle$. These are concatenated to form the bigram index, $blue shoes = |101100\rangle$.

The QCBM uses parametric quantum circuits to generate different superpositions. The training challenge is then to find the proper set of parameters that lead to the generation of the specific superposition that represents the target joint probability distribution. For this, a Simultaneous Perturbation Stochastic Approximation (SPSA) optimizer (Spall, 1998) was implemented which optimizes the parametric quantum circuit, minimizing the difference between the generated distribution and the target distribution using the Kullback–Leibler (KL) divergence as the cost function. Sample results are shown in Figure 5. The initial results were disappointing (middle column) — while finding some peaks in the distribution, the optimizer entirely missed others, leading to a poor KL-divergence score of 1.131. Our hypothesis was that the target distribution was too discrete or 'spiky' to be fitted effectively in this way. Further experiments were carried out that *added* noise to the distribution and tried to fit the noisy version. Perhaps surprisingly, this worked considerably better — the optimizer was able to fit most of the spikes in the distribution, at the cost of giving some weight to other elements, achieving a much better KL-divergence of 0.211.

This result is curiously analogous to the use of smoothing in n-gram models. It is often observed that language produces new words and phrases that have not been seen before: this is what is meant when language is said to be 'generative'. To accommodate this, language models are adapted to give non-zero







Model trained with noise

Figure 5: Attempts to fit the target distribution, with and without introducing noise. The goodness-of-fit is measured by Kullback-Liebler (KL) divergence, $D_{\text{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)}\right)$, where P is the learned distribution and Q is the target distribution.

probabilities even to phrases that have never been encountered in training, and there are several traditional ways of doing this (Manning and Schütze, 1999, Ch 6).

In a simple demonstration, after training, the model was used to generate a few new phrases by random sampling. Results included some plausible example like *black suit*, and occasionally misfits such as *ripe shirt*. A followup experiment is planned to develop such a model with a considerably larger dataset, and to measure generative performance using a traditional measure such as perplexity (Manning and Schütze, 1999, Ch 2).

5 Ambiguity Resolution in Composition

The ability to choose appropriate interpretations for ambiguous words is a hallmark of human language capability. Much of the time, we guess meanings quickly and correctly, but various experiments have also shown that in some situations, several meanings are not only available but cognitively activated, causing and confusion and delay in human language responses (Aitchison, 2002, Ch 19). ²

Word-sense discrimination and disambiguation were among the early successes of word vector models during the 1990's, built using dimension-reduction of term-document or term-term cooccurrence matrices (Schütze, 1998). This was followed by the use of orthogonal projection as a logical operator for isolating particular senses of ambiguous words, confirming that a semantic vector for an ambiguous word can be analyzed as a sum of vectors for difference senses, which can sometimes be recovered from the geometry of the vector space (Widdows, 2004, Ch 7).

The ambition of combining distributional word vectors with compositional operations that model grammatical structures led to successful experiments with matrices as operators — for example, where nouns are modelled as vectors, adjectives can be modelled more effectively as matrices acting upon the noun vectors Baroni and Zamparelli (2010). Putting such insights together enabled comprehensive systems for distributional compositional semantics to be devised (Coecke et al., 2010; Baroni et al., 2014), leading to many successful experiments (for a more thorough summary see Widdows et al. (2021)).

²Sometimes ambiguity resolution is so effortless that we underestimate the prevalence of ambiguity itself, which takes many forms. For example, English has considerable syntactic ambiguity — one large sample found that 57% of the 400 most common English verbs are also nouns (Widdows, 2003b), but we rarely notice this. As noted by Baroni et al. (2014), "Nearly all papers in formal semantics start from the assumption that all the words in the examples under study have been neatly disambiguated before being combined. But this is not an innocent or easy assumption."

Table 2: A toy domain model with column vectors for nouns and matrices for transitive verbs

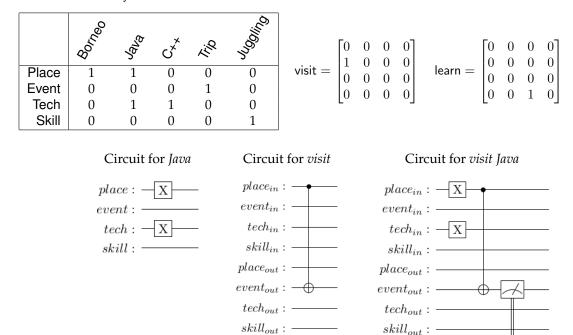


Figure 6: Toy circuit examples showing how the phrase visit Java selects the place meaning

 $event_{seen} : \frac{}{}$

As noted by authors including Baroni and Zamparelli (2010), the use of matrices to operate on vectors supports disambiguation as different components of meaning are selected in the compositional process. For example, we might learn that *red* means something closer to *orange* in 'red hair', something closer to *purple* in 'red wine', and something closer to *urgent* in 'red alert'.

To demonstrate such a process on quantum computer, we designed the example domain shown in Table 2. There are only four dimensions, representing the broad areas *place*, *event*, *technology*, *skill*, and 5 nouns — one belonging to each dimension, and the noun *Java* which could refer to a place or a technology. Consider now the transitive verbs *visit* and *learn*. When we say "I visited Borneo", this indicates that an event happened — so the verb *visit* takes places as input and creates events as output. When we say "I learned C++", this indicates that a skill was acquired — so the verb *learned* in this case takes technologies as input and creates skills as output. It is easy to write down matrices that perform these operations (Table 2, right hand side).

Consider now the action of these matrices on the ambiguous word *Java*. Both the *place* and the *tech* dimensions are activated with this input, but the verb *visit* simply ignores the *tech* part of the input, whereas the verb *learn* ignores the *place* part. In the process of composition, the meanings that naturally combine with one another are preserved, and those that do not are discarded.

This process was then modelled using the quantum circuits in Figure 6 — for example, the verb *visit* is implemented by connecting the input *place* qubit to the output *event* qubit. The quantum circuit is actually more complicated mathematically than the small matrices of Table 2 — for example, the state of the four input qubits would be represented in $(\mathbb{C}^2)^{\otimes 4} \cong \mathbb{C}^{16}$. These interactions would have to be considered when modelling a larger domain, if only because the approach in Figure 6 would soon run out of qubits.

Nonetheless, an experiment with these circuits on the 11-qubit IonQ machine gave the expected results — measuring the output bits for the *visit Java* or the corresponding *learn Java* circuits shows that the former activates the *event* output and the latter activates the *skill* output. Ambiguity is an example of a problem that is crucial to language understanding, to which the mathematical tools of vectors and matrices have been successfully applied, and the circuits of Figure 6 provide an example of how such processes can be implemented on quantum computers.

6 Related Work

These experiments and results come in the midst of much progress in quantum machine learning and AI. Until recently, most work in quantum algorithms was theoretical in the sense that these algorithms could not be run on quantum hardware. This has changed rapidly, and in the 2020's, new results from quantum computers in artificial intelligence (AI) are being published almost every month, with applications including probabilistic reasoning (Borujeni et al., 2021) financial modeling (Johri et al., 2021), and image classification Wang et al. (2021); Johri et al. (2022). Similar questions arise across these areas, partly because the availability of some of the key quantum properties like indeterminacy, interference, and entanglement pose the question "How can these mathematical properties help to model a given situation?"

The most directly comparable research to that presented here is the compositional quantum approach released in the lambeq package (Kartsaklis et al., 2021). This is the ongoing culmination of a research program outlined by Coecke et al. (2010), which is often referred to as the DisCoCat ('distributional compositional categorical') model. This combines the use of distributional semantic vectors with syntactic parsing, and partly follows the tradition of logical semantics in the sense that the semantic representation is derived from first finding the syntactic or grammatical structure of a sentence. This approach to meaning is natural from the point of view of linguistic theory, which often consider sentences as fundamental units of language and require grammaticality. An approach based on information retrieval (following for example Van Rijsbergen (2004)) is more likely to start with individual words, and move from there to pairs of words (Sordoni et al., 2013), and the examples in this paper have more obviously drawn on this approach. Given the huge range of language challenges in computing, and pace of development in both language processing and quantum computing, we expect that the next few years will witness many more contributions and combinations.

7 Conclusions and Next Steps

The experiments and demonstrations presented in this paper show that NLP tasks can be performed on quantum computers, which is important in itself — quantum information processing is being demonstrated on quantum hardware, not just in theory. The practical applications are so far small-scale, and it is still challenging to find consumer-facing NLP requirement that would be better served today by quantum than classical hardware. The main dependency here is hardware: because quantum memory capacity grows exponentially with the number of qubits, 10 qubits can represent a kilobyte, whereas 30 qubits can represent a gigabyte (roughly speaking — this compares the number of variables, not their expressivity or access patterns, and classical memory is still much more reliable). This is enough to explore much larger datasets, and to look for useful intermediate-scale applications. Though much of the focus in NLP research

in recent years has been on larger and larger models, many commercial systems involve much smaller curated datasets, such as the intents used by dialog systems or ticket classifications used by customer service tracking systems. With a dedicated focus on building useful products, we believe it is possible for quantum NLP to help with practical medium-term goals, and this can show the way to more general intelligent systems.

References

- Aitchison, J. (2002). Words in the Mind: An Introduction to the Mental Lexicon. Blackwell, 3rd edition.
- ANIS, M. S., Abby-Mitchell, Abraham, H., AduOffei, Agarwal, R., Agliardi, G., and other authors (2021). Qiskit: An open-source framework for quantum computing.
- Baroni, M., Bernardi, R., Zamparelli, R., et al. (2014). Frege in space: A program for compositional distributional semantics. *Linguistic Issues in language technology*, 9(6):5–110.
- Baroni, M. and Zamparelli, R. (2010). Nouns are vectors, adjectives are matrices: Representing adjectivenoun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Birkhoff, G. and von Neumann, J. (1936). The logic of quantum mechanics. *Annals of Mathematics*, 37:823–843.
- Borujeni, S. E., Nannapaneni, S., Nguyen, N. H., Behrman, E. C., and Steck, J. E. (2021). Quantum circuit representation of bayesian networks. *Expert Systems with Applications*, 176:114768.
- Bradley, T.-D. (2020). At the Interface of Algebra and Statistics. PhD thesis, City University of New York.
- Busemeyer, J. R. and Bruza, P. D. (2012). *Quantum models of cognition and decision*. Cambridge University Press.
- Coecke, B., de Felice, G., Meichanetzidis, K., and Toumi, A. (2020). Foundations for near-term quantum natural language processing. *arXiv preprint*, arXiv:2012.03755.
- Coecke, B., Sadrzadeh, M., and Clark, S. (2010). Mathematical foundations for a compositional distributional model of meaning. *CoRR*, abs/1003.4394.
- Cohen, T., Widdows, D., Vine, L. D., Schvaneveldt, R., and Rindflesch, T. C. (2012). Many paths lead to discovery: Analogical retrieval of cancer therapies. In *Sixth International Symposium on Quantum Interaction*.
- Cooke, H. P. and Treddenick, H., editors (1938). *Aristotle: Categories, On Interpretation, Prior Analytics*. Loeb Classical Library. Harvard University Press.
- Dirac, P. (1930). *The Principles of Quantum Mechanics*. Clarendon Press, Oxford, 4th edition, 1958, reprinted 1982 edition.
- Ganter, B. and Wille, R. (1999). Formal Concept Analysis: Mathematical Foundations. Springer.
- Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media.

- IonQ Aria (2022). IonQ Aria furthers lead as world's most powerful quantum computer. https://ionq.com/news/february-23-2022-ionq-aria-furthers-lead, accessed 2022-05-28.
- Johri, S., Debnath, S., Mocherla, A., Singk, A., Prakash, A., Kim, J., and Kerenidis, I. (2021). Nearest centroid classification on a trapped ion quantum computer. *npj Quantum Information*, 7(1):1–11.
- Johri, S., Zhu, E., Bacon, D., Esencan, M., Kim, J., Muir, M., Murgai, N., Nguyen, J., Pisenti, N., Schouela, A., et al. (2022). Generative quantum learning of joint probability distribution functions. *Bulletin of the American Physical Society*.
- Kartsaklis, D., Fan, I., Yeung, R., Pearson, A., Lorenz, R., Toumi, A., de Felice, G., Meichanetzidis, K., Clark, S., and Coecke, B. (2021). lambeq: An Efficient High-Level Python Library for Quantum NLP. *arXiv* preprint arXiv:2110.04236.
- Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., and Brown, D. (2019). Text classification algorithms: A survey. *Information*, 10(4):150.
- Lorenz, R., Pearson, A., Meichanetzidis, K., Kartsaklis, D., and Coecke, B. (2021). QNLP in practice: Running compositional models of meaning on a quantum computer. *arXiv* preprint arXiv:2102.12846.
- Manning, C. D. and Schütze, H. (1999). Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge, Massachusetts.
- Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint*, arXiv:1301.3781.
- Nielsen, M. A. and Chuang, I. (2002). *Quantum computation and quantum information*. American Association of Physics Teachers.
- Orrell, D. (2020). *Quantum Economics and Finance: An Applied Mathematics Introduction*. Panda Ohana Publishing.
- Orrell, D. and Houshmand, M. (2022). Quantum propensity in economics. *Frontiers in Artificial Intelligence*, 4.
- Piedeleu, R., Kartsaklis, D., Coecke, B., and Sadrzadeh, M. (2015). Open system categorical quantum semantics in natural language processing. *arXiv* preprint arXiv:1502.00831.
- Pothos, E. M. and Busemeyer, J. R. (2022). Quantum cognition. Annual review of psychology, 73:749–778.
- Schrödinger, E. (1935). Discussion of probability relations between separated systems. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 31, pages 555–563. Cambridge University Press.
- Schütze, H. (1998). Automatic word sense discrimination. Computational Linguistics, 24(1):97–124.
- Shannon, C. (1948). A mathematical theory of communication. *Bell system technical journal*, 27:379–423, 623–656.
- Sordoni, A., Nie, J.-Y., and Bengio, Y. (2013). Modeling Term Dependencies with Quantum Language Models for IR. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 653–662.

- Spall, J. C. (1998). An overview of the simultaneous perturbation method for efficient optimization. *Johns Hopkins apl technical digest*, 19(4):482–492.
- Van Rijsbergen, C. J. (2004). The Geometry of Information Retrieval. Cambridge University Press.
- Wang, K., Xiao, L., Yi, W., Ran, S.-J., and Xue, P. (2021). Experimental realization of a quantum image classifier via tensor-network-based machine learning. *Photonics Research*, 9(12):2332–2340.
- Widdows, D. (2003a). A mathematical model for context and word-meaning. In *International and Interdisci- plinary Conference on Modeling and Using Context*, pages 369–382. Springer.
- Widdows, D. (2003b). Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In *Proceedings of Human Language Technology / North American Chapter of the Association for Computational Linguistics*, Edmonton, Canada.
- Widdows, D. (2004). Geometry and Meaning. CSLI Publications.
- Widdows, D. (2022). Nonlinear addition of qubit states using entangled quaternionic powers of single-qubit gates. *arXiv preprint arXiv:2204.13787*.
- Widdows, D., Kitto, K., and Cohen, T. (2021). Quantum mathematics in artificial intelligence. *Journal of Artificial Intelligence Research*, 72:1307–1341.