Towards Quantum Language Models

Ivano Basile
Scuola Normale Superiore, Pisa, Italy
ivano.basile@sns.it

Fabio Tamburini
FICLIT - University of Bologna, Italy
fabio.tamburini@unibo.it

Abstract

This paper presents a new approach for building Language Models using the Quantum Probability Theory, a Quantum Language Model (QLM). It mainly shows that relying on this probability calculus it is possible to build stochastic models able to benefit from quantum correlations due to interference and entanglement. We extensively tested our approach showing its superior performances, both in terms of model perplexity and inserting it into an automatic speech recognition evaluation setting, when compared with state-of-the-art language modelling techniques.

1 Introduction

Quantum Mechanics Theory (QMT) is one of the most successful theories in modern science. Despite its effectiveness in the physics realm, the attempts to apply it in other domains remain quite limited, excluding, of course, the large quantity of studies regarding Quantum Information Processing on quantum computers.

Only in recent years some scholars tried to embody principles derived from QMT into their specific fields, for example, by the Information Retrieval community (Zuccon et al., 2009; Melucci and van Rijsbergen, 2011; González and Caicedo, 2011; Melucci, 2015) and in the domain of cognitive sciences and decision making (Khrennikov, 2010; Busemeyer and Bruza, 2012; Aerts et al., 2013). In the machine learning field (Arjovsky et al., 2016; Wisdom et al., 2016; Jing et al., 2017) have used unitary evolution matrices for building deep neural networks obtaining interesting results, but we have to observe that their works do not adhere to QMT and use unitary evolution operators in a way not allowed by QMT. In recent years, also the Natural Language Processing (NLP) community started to look at QMT with interest and some studies using it have already been presented (Blacoe et al., 2013; Liu et al., 2013; Tamburini, 2014; Kartsaklis et al., 2016).

Language models (LM) are basic tools in NLP used in various applications, such as Automatic Speech Recognition (ASR), machine translation, part-of-speech tagging, etc., and were traditionally modeled by using N-grams and various smoothing techniques. Among the dozen of tools for computing N-gram LM, we will refer to CMU-SLM (with Good-Turing smoothing) (Clarkson and Rosenfeld, 1997) and IRSTLM (with Linear Witten-Bell smoothing) (Federico et al., 2008); the latter is the tool used in Kaldi (Povey et al., 2011b), one of the most powerful and used open-source ASR package that we will use for some of the experiments presented in the following sections.

In recent years new techniques from the Neural Networks (NN) domain have been introduced in order to enhance the performances of such models. Elman recurrent NN, as used in the RNNLM tool (Mikolov et al., 2010, 2011), or Long Short-Term Memory NN, as in the tool LSTMLM (Soutner and Müller, 2015), produce state-of-the-art performances for current language models.

This paper presents a different approach for building LM based on quantum probability theory. Actually, we present a QLM applicable only to problems defined on a small set of different tokens. This is a "proof-of-concept" study and our main aim is to show the potentialities of such approach rather than building a complete application for solving this problem for any setting.

The paper is organized as follows: we provide background on Quantum Probability Theory in Section 2 followed by the description of our proposed Quantum Language Model in Section 3. We then discuss some numerical issues mainly related

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1840–1849
Copenhagen, Denmark, September 7–11, 2017.

to the optimisation procedure in Section 4, and in Section 5 we present the experiments we did to validate our approach. In Section 6 we discuss our results and draw some provisional conclusions.

2 Quantum Probability Theory

In QMT the state of a system is usually described, in the most general case, by using density matrices over an Hilbert space $\mathcal{H}$. More specifically, a *density matrix* ρ is a positive semidefinite Hermitian matrix of unit trace, namely $\rho^\dagger = \rho, \quad \mathrm{Tr}(\rho) = 1$, and it is able to encode all the information about the state of a quantum system[1].

The measurable quantities, or *observables*, of the quantum system are associated to Hermitian matrices O defined on $\mathcal{H}$. The axioms of QMT specify how one can make predictions about the outcome of a measurement using a density matrix:

- the possible outcomes of a projective measurement of an observable O are its eigenvalues $\{\lambda_j\}$;
- the *probability* that the outcome of the measurement is λ_j is $P(\lambda_j) = \mathrm{Tr}(\rho\Pi_{\lambda_j}) = \mathrm{Tr}(\Pi_{\lambda_j}\rho)$, where Π_{λ_j} is the projector on the eigenspace of O associated to λ_j. Note that in the following we will use some properties of these kind of measurements, namely $\Pi^\dagger_{\lambda_j} = \Pi_{\lambda_j}$ and $\Pi^2_{\lambda_j} = \Pi_{\lambda_j}$;
- after the measurement the system state *collapses* in the following fashion: if the outcome of the measurement was λ_j, the collapse is
$$\rho' = \frac{\Pi_{\lambda_j}\rho\Pi_{\lambda_j}}{\mathrm{Tr}(\Pi_{\lambda_j}\rho\Pi_{\lambda_j})}$$
where the denominator is needed for trace normalization;
- time evolution of states using a fixed time step is described by a *unitary matrix* U over $\mathcal{H}$, i.e. $U^\dagger U = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Given a state ρ_t, at a specific time t, the system evolution without measurements modifies the state as:
$$\rho_{t+1} = U\rho_t U^\dagger.$$

See for example (Nielsen and Chuang, 2010) or (Vedral, 2007) for a complete introduction on QPT.

[1] $\dagger$ marks the conjugate transpose of a vector/matrix and $\mathrm{Tr}(\cdot)$ is the trace of a matrix.

3 Quantum Language Models

In this section we describe our approach to build QLM that can compute probabilities for the occurrence of a sequence $\mathbf{w} = (w_1, w_2, ..., w_n)$ of length n, composed using N different symbols, the vocabulary containing all the words in the model, i.e. for every symbol w in the sequence $w \in \{0, ..., N-1\}$. We define a set of orthogonal N-dimensional vectors $\{\mathbf{e}_w : w \in \{0, ..., N-1\}\}$, spanning the complex space $\mathcal{H} = \mathbb{C}^N$; to measure the probability of a symbol w, collapsing the state over the space spanned by $\mathbf{e}_w$, we use the projector $\Pi_w = \mathbf{e}_w\mathbf{e}_w^\dagger$. Note that all the words in the vocabulary have been encoded as numbers corresponding to the N dimensions of the vector space $\mathcal{H}$.

Our method is sequential, from QMT point of view, in the sense that we use a quantum system that produces a single symbol upon measurement.

The basic idea is that the probabilistic information for a given sequence $\mathbf{w} = (w_1, w_2, ..., w_n)$ is encoded in the density matrix that results from the following process:

- **Inititalisation**

  Cond.Prob.: $P(w_1; \rho_0, U) = \mathrm{Tr}(\rho_0\Pi_{w_1})$

  Projection: $\rho'_1 = \frac{\Pi_{w_1}\rho_0\Pi_{w_1}}{\mathrm{Tr}(\Pi_{w_1}\rho_0\Pi_{w_1})}$

  Evolution: $\rho_1 = U\rho'_1U^\dagger$

- **Recurrence** $(i = 2, .., n)$

  Cond.Prob.: $P(w_i|w_1, ..., w_{i-1}; \rho_0, U) = \mathrm{Tr}(\rho_{i-1}\Pi_{w_i})$

  Projection: $\rho'_i = \frac{\Pi_{w_i}\rho_{i-1}\Pi_{w_i}}{\mathrm{Tr}(\Pi_{w_i}\rho_{i-1}\Pi_{w_i})}$

  Evolution: $\rho_i = U\rho'_iU^\dagger$

- **Termination**

$$P(\mathbf{w}|\rho_0, U) = P(w_1; \rho_0, U) \cdot \prod_{i=2}^{n} P(w_i|w_1, ..., w_{i-1}; \rho_0, U)$$

The total probability $P(\mathbf{w}|\rho_0, U)$ for the given sequence is thus obtained, in the termination step, by multiplying the conditional probability $P(w_i|w_1, ..., w_{i-1}; \rho_0, U)$ for each word in the sequence.

We then use the initial density matrix ρ_0 and the time evolution unitary matrix U as parameters to optimise the *perplexity* Γ, evaluated on a training

corpus of sequences S,

$$\Gamma(\rho_0, U) = \exp\Big(-\frac{1}{C}\sum_{\mathbf{w}\in S}\log P(\mathbf{w}|\rho_0, U)\Big)$$

which quantifies the uncertainty of the model. C is the number of tokens in the corpus.

Minimising Γ is equivalent of learning a model by fixing all the model parameters, a typical procedure in the machine learning domain.

3.1 Ancillary system

The problem with this setup is that the 'quantum effects' are completely washed out by the measurements on the system by using projectors. The resulting expression for the probability $P(\mathbf{w}|\rho_0, U)$ for a sequence $\mathbf{w}$ is identical to that obtained using a classical Markov model.

To solve this issue, our approach is to *avoid the complete collapse* of the state after each symbol measurement using a common technique in QMT: we introduce an *ancillary system* described by a fictitious D-dimensional Hilbert space, $\mathcal{H}_{ancilla} = \mathbb{C}^D$, and we couple the original system to the ancillary system. The resulting DN-dimensional Hilbert space is

$$\mathcal{H}_2 \equiv \mathcal{H}_{ancilla} \otimes \mathcal{H} = \mathbb{C}^{DN}$$

where $\otimes$ denotes the Kronecker product for matrices and D can be seen as a free hyper-parameter of the model. On this new space the projectors are now given by $\Pi_w^{(2)} = \mathbf{I}_D \otimes \Pi_w$, where $\mathbf{I}_D$ is the D-dimensional identity matrix.

The advantage of using this method is that the time evolution for the coupled system creates non-trivial correlations between the two entangled systems such that measuring and collapsing the symbol state keeps some information about the whole sequence stored in the ancillary part of the state. This information is then reshuffled into the symbol state via time evolution, resulting in a 'memory effect' that takes the whole sequence of symbols into account, thereby extending the idea behind the N-grams approach. Larger D values will results in more memory of this system and, of course, in a larger number of parameters to learn.

3.2 System evolution

We need to specify the system evolution for our coupled system. The simplest approach is to use a unitary $DN \times DN$ matrix U that acts on the entangled Hilbert space as shown before; it can be specified by $(DN)^2$ real parameters with a suitable parametrization (Spengler et al., 2010) that ensures the unitarity of U. However, in our preliminary experiments this approach resulted in an insufficient 'memory' capability for the QLM and in a very complex and slow minimisation procedure.

A different approach could be introduced by using a specific unitary matrix for each word, but this would lead to an enormous amount of parameters to learn with the optimization procedure.

There are a lot of techniques in NLP to represent single words with dense vectors (see for example (Mikolov et al., 2013) for the so called *word embeddings*). Following this idea, we can represent every symbol in our system with a specific p-dimensional vector trained using one of the available techniques $w \mapsto (\alpha_1(w), ..., \alpha_p(w))$ or fixed randomly.

We then work with a set of p $DN \times DN$ unitary matrices $\mathbf{U} = (U_1, ..., U_p)$, one for each component of the word vector, that are used to dynamically build a different system evolution matrix for each word in this way:

$$V(w) \equiv \prod_{i=1}^{p} U_i^{\alpha_i(w)}$$

This results in $p(DN)^2$ complex or $2p(DN)^2$ real parameters to be learned.

Essentially, we treat the words in our problem in different ways: the evolution operator for each word $V(w)$ is build by using a combination of the operators $\mathbf{U}$ defined for each word-vector component, while, considering the system projection, we treat each word as one basis vector for the space $\mathcal{H}$.

Note that the choice to use a set $\{V(w)\}$ of operators, one for each word w, does not violate the linearity of quantum mechanics: let K be the quantum operation

$$K(\rho) = \sum_w V(w)\Pi_w^{(2)}\rho\,\Pi_w^{(2)}V^\dagger(w)$$

defined using projectors and evolution matrices. Then K is a valid (i.e. a Completely Positive Trace-preserving) evolution map that exactly reproduces our results in the sequence of evolutions and collapses.

The number of evolutionary operators is a trade-off: as we said before, defining only one operator U resulted in a poor performance of the

proposed method in all the relevant experiments, while defining an operator for each word would produce too many parameters to be learned. The trade-off that we chose is to use one operator for each word-vector component, and build the set $\{V(w)\}$ from them as described above while preserving unitarity.

With regard to the initial density matrix ρ_0, we have to define it combining the initial density matrix of our system, ρ_0^s, and the initial density matrix of the ancilla, ρ_0^a. We defined ρ_0^s as a diagonal $N \times N$ matrix containing the classical Maximum Likelihood probability Estimation to have a specific symbol at the first sequence position:

$$\rho_0^s = \frac{1}{|S|} \sum_{\mathbf{w} \in S} \Pi_{w_1}$$

where S is again the set of all sequences in the training set and w_1 is the first word in each sequence $\mathbf{w}$. With regard to the ancilla system we do not know anything about it and thus we have to define ρ_0^a as the $D \times D$ diagonal matrix

$$\rho_0^a = \frac{\mathbf{I}_D}{\mathrm{Tr}(\mathbf{I}_D)} \, .$$

Consequently we can define ρ_0 as

$$\rho_0 = \rho_0^a \otimes \rho_0^s \, .$$

3.3 The final model

Putting all the ingredients together, we can finally write down the formula for the probability $P(\mathbf{w}|\rho_0, \mathbf{U})$ for a sequence $\mathbf{w}$ in the QLM specified by ρ_0 and $\mathbf{U}$. The product of conditional probabilities simplifies because of the normalising denominators added at each collapse and time evolution step. The result is:

$$P(\mathbf{w}|\rho_0, \mathbf{U}) = \mathrm{Tr}(\Pi_{w_n}^{(2)} ... V^\dagger(w_2) \Pi_{w_2}^{(2)} V^\dagger(w_1) \Pi_{w_1}^{(2)} \rho \Pi_{w_1}^{(2)} V(w_1) \Pi_{w_2}^{(2)} V(w_2) ... \Pi_{w_n}^{(2)}) \quad (1)$$

Using the fact that projectors have many zero entries one can also re-express this trace of the product of $DN \times DN$ matrices in terms of the trace of the product of $D \times D$ matrices. The formula for $P(\mathbf{w}|\rho_0, \mathbf{U})$ then simplifies to our final result

$$P(\mathbf{w}|\rho_0, \mathbf{U}) = \mathrm{Tr}(T^\dagger R T) \quad (2)$$

where the matrices R and T are defined as follows:

- in terms of entries $R_{i,j}$ with indices $i, j = 0, ..., D-1$, the matrix R is given by

  $$R_{i,j} = [\rho_0]_{Ni+w_1, Nj+w_1} .$$

  Note that only the value of first symbol in the sequence, w_1, enters in the expression. This is to be expected since R derives from the initial density matrix ρ_0;

- analogously, the matrix T that encodes the chain of combined collapses and time evolutions is given by the product $T = T^{(2)} T^{(3)} ... T^{(n)}$, where the matrices $T^{(k)}$ are given in entries, with indices $i, j = 0, ..., D-1$, by

  $$T_{i,j}^{(k)} = [V(w_{k-1})]_{Ni+w_{k-1}, Nj+w_k} .$$

  These matrices can be pre-calculated for every pair of the involved symbols, so that the calculation of $P(\mathbf{w}|\rho_0, \mathbf{U})$ for all the sequences will be very fast.

The detailed calculation for obtaining the equation (2) can be found in the supplementary material.

4 Optimisation and Numerical Issues

In order to optimise the parameters $\mathbf{U}$ we numerically minimise the perplexity Γ computed on a given training corpus of sequences S. This requires that the matrices $\mathbf{U}$ remain strictly unitary at every step of the minimisation procedure and it can be accomplished in various ways.

The most straightforward way is to employ an explicit parametrization for unitary matrices, as was done in (Spengler et al., 2010). Due to the transcendental functions employed in this parametrisation, this approach resulted in a functional form for Γ that has proven to be very challenging to minimise efficiently in our experiments.

A more elegant and efficient approach is to consider the entries of $\mathbf{U}$ as parameters (thereby ensuring a polynomial functional form for Γ) and to employ techniques of differential geometry to keep the parameters from leaving the unitary subspace at each minimisation step. This can be done using a modification of the approach outlined in (Tagare, 2011) that considers the unitary matrices subspace as a manifold, the *Stiefel manifold* $\mathrm{U}(DN)$. It is then possible to project the gradient ∇f of a generic function $f(M)$ of the matrix variable M on the tangent space of the Stiefel manifold and build a line search algorithm that sweeps

out curves on this manifold so that at each point the parameters are guaranteed to form a unitary matrix.

In our case we have multiple unitary matrices $\mathbf{U} = (U_1, ..., U_p)$. This simply results having curves defined on $\mathrm{U}(DN)^p$, parametrised by a p-dimensional vector of $DN \times DN$ unitary matrices.

4.1 Formula for the gradient

To implement the curvilinear search method described in (Tagare, 2011) one needs an expression for the gradient $\mathbf{G} = (G_1, ..., G_p)$ of the probability function. This gradient is organised in a p-dimensional vector of $DN \times DN$ matrices, such that the component G_j is obtained by computing the matrix derivative of $P(\mathbf{w}|\rho_0, \mathbf{U})$ with respect to U_j either analytically or by applying some numerical estimate of the gradients, for example by using finite differences. The latter method, when working with thousands or millions of variables can be very time consuming and, usually, an explicit analytic formula for the gradient accelerates considerably all the required processing.

A lengthy analytic computation results in an explicit result. Firstly, we introduce the following objects:

- The *spectral decomposition* of U_j, given by $U_j = S_j D_j S_j^\dagger$, guaranteed to exist by the spectral theorem. S_j is unitary and the diagonal matrix D_j contains the eigenvalues $(u_{j1}, ..., u_{jDN})$ of U_j, $j = 1, ..., p$.

- The $DN \times DN$ matrices $C_j(\alpha)$ defined, in entries, by

$$[C_j(\alpha)]_{ab} = \frac{\overline{u_{ja}}^\alpha - \overline{u_{jb}}^\alpha}{\overline{u_{ja}} - \overline{u_{jb}}} \quad \text{if} \quad u_{ja} \neq u_{jb}$$

$$[C_j(\alpha)]_{ab} = \alpha \overline{u_{ja}}^{\alpha-1} \quad \text{if} \quad u_{ja} = u_{jb}$$

  where $\overline{u}$ is the complex conjugate of u.

- The $D \times DN$ matrices Q_k given in entries by

$$(Q_k)_{jA} = \delta_{Nj+w_k, A}$$

  where $j = 0, ..., D-1, \quad A = 0, ..., DN-1$.

- The *lesser* and *greater products* associated to the construction of system evolution matrices

$$V^{<j}(w) = \prod_{i=1}^{j-1} U_i^{\alpha_i(w)}$$

$$V^{>j}(w) = \prod_{i=j+1}^{n} U_i^{\alpha_i(w)}.$$

With these ingredients, the resulting formula for the components G_j of the gradient is

$$G_j = 2S_j \sum_{k=2}^{n} \Big\{ \Big[S_j^\dagger \Big(V^{<j}(w_{k-1})^\dagger Q_{k-1}^T \Big(\prod_{l=2}^{k-1} T^{(l)} \Big)^\dagger R T \Big(\prod_{l=k+1}^{n} T^{(l)} \Big)^\dagger Q_k V^{>j}(w_{k-1})^\dagger \Big) S_j \Big] \cdot C_j(\alpha_j(w_{k-1})) \Big\} S_j^\dagger \tag{3}$$

where $\cdot$ denotes the element-wise matrix product. Again, all the detailed calculations for obtaining the analytic expression (3) for the gradient G_j can be found in the supplementary material.

Using Tagare's method we can project the gradient onto the Stiefel manifold and build a curvilinear search algorithm for the minimisation.

To achieve this aim, Tagare proposed an Armijo-Wolfe line search inserted into a simple gradient descent procedure. We developed an extension of this algorithm combining the minimization over the Steifel manifold technique with a Moré-Thuente (1994) line search and a Conjugate Gradient minimisation algorithm that uses the Polak-Ribière method for the combination of gradients and search directions (Nocedal and Wright, 2006). All the experiments presented in the next section were performed using these methods.

The minimisation uses random mini-batches that increase their size during the training: they start with approximately one tenth of the training set dimension and increase to include all the instances using a parametrised logistic function. As stopping criterion we used the minimum of the perplexity function over the validation set as suggested in (Bengio, 2012; Prechelt, 2012) for other machine learning techniques.

5 Experiments and Results

5.1 Data

The TIMIT corpus is a read speech corpus designed to provide speech data for acoustic-phonetic studies and for the development and evaluation of automatic speech recognition systems (Garofolo et al., 1990). It contains broadband recordings of 630 speakers of eight major dialects

of American English and includes time-aligned orthographic, phonetic and word transcriptions as well as a 16-bit, 16 kHz speech waveform file for each utterance.

In the speech community, the TIMIT corpus is the base for a standard phone-recognition task with specific evaluation procedures described in detail in (Lopes and Perdigao, 2011). We stick completely to this evaluation to test the effectiveness of our proposed model adopting, among the other procedures, the same splitting between the different data sets: the training set contains 3696 utterances (140225 phones), the validation set 400 utterances (15057 phones) and the test set 192 utterances (7215 phones).

5.2 Evaluation Results

We tested the proposed model by setting up two different evaluations: the first is an intrinsic evaluation of LM performances in terms of global perplexity on the TIMIT testset; the second is an extrinsic evaluation in which we replace the LM tools provided with the Kaldi ASR toolkit (Povey et al., 2011b) with our model in order to check the final system performances in a phone-recognition task and comparing them with the other state-of-the-art LM techniques briefly introduced in Section 1.

5.2.1 Intrinsic evaluation

The first experiment consisted in an evaluation of models perplexity (PPL) on the TIMIT testset. We compared the QLM model with two N-gram implementations, namely CMU-SLM (Clarkson and Rosenfeld, 1997) and IRSTLM (Federico et al., 2008), and two recurrent NN models able to produce state-of-the-art results in language modelling, the RNNLM (Mikolov et al., 2010, 2011) and the LSTMLM (Soutner and Müller, 2015) packages.

Table 1 shows the results of the intrinsic evaluation. With regard to RNNLM and LSTMLM results, only the best hyper-parameters combination after a lot of experiments, optimizing them on the validation set, has been inserted into the Table.

With regard to QLM, all the presented experiments are based on artificial word vectors produced randomly using values from the set $\{-1, 0, 1\}$ instead of real word embeddings. Every word vector is different from the others and we decided not to use real embeddings in order to test the core QMT method without adding the contextual information, contained in word embeddings, that could have helped our approach to obtain better performances, at least in principle.

Model	Parameters	PPL
CMU-SLM (Good-Turing smoothing)	2-gram	15.49
	3-gram	14.28
	4-gram	15.62
	5-gram	17.33
IRSTLM (linear Witten-Bell smoothing)	2-gram	15.47
	3-gram	14.07
	4-gram	15.55
	5-gram	17.53
RNNLM	280 neurons	13.32
LSTMLM	25 neurons, 1 layer	13.17
QLM	N=48, p=4, D=10	13.44
	N=48, p=4, D=20	**13.15**
	N=48, p=4, D=30	**13.10**
	N=48, p=4, D=40	**12.99**

Table 1: Perplexity (PPL) of the tested language-modelling techniques on the TIMIT testset. All the QLM results in bold face are better than the other systems we tested.

5.2.2 Extrinsic evaluation

The "TIMIT recipe" contained in the Kaldi distribution[2] reproduces exactly the same evaluation settings described in (Lopes and Perdigao, 2011) for a phone recognition task based on this corpus. Moreover, Kaldi provides some n-best rescoring scripts that apply RNNLM hypothesis rescoring and interpolate the results with the standard N-gram model results used in the evaluation. We slightly modified these scripts to work with LSTMLM and QLM in order to test different models using the same setting. This allowed us to replace the LM used in Kaldi and experiment with all the systems evaluated in the previous section.

Table 2 outlines the results we obtained replacing the LM technique into Kaldi ASR package w.r.t. the different ASR systems that the TIMIT recipe implements. These systems are built on top of MFCC, LDA, MLLT, fMLLR with CMN[3] features (see (Povey et al., 2011b; Rath et al., 2013) for all acronyms references and a complete feature

[2] https://github.com/kaldi-asr/kaldi

[3] MFCC: Mel-Frequency Cepstral Coefficients; LDA: Linear Discriminant Analysis; MLTT: Maximum Likelihood Linear Transform; fMLLR: feature space Maximum Likelihood Linear Regression; SAT: Speaker Adapted Training, i.e. train on fMLLR-adapted features; CMN: Cepstral Mean Normalization.

or recipe descriptions).

For this extrinsic evaluation we used the best models we obtained in the previous experiments interpolating their log-probability results for each utterance with the original bigram (or trigram) log-probability using a linear model with a ratio 0.25/0.75 between the original N-gram LM and the tested one as suggested in the standard Kaldi rescoring script. For this test we rescored the 10,000-best hypothesis.

We have to say that in this experiment we were not trying to build the best possible phone recogniser, but simply to compare the relative performances of the analysed LM techniques showing the effectiveness of QLM when used in a real application. Thus absolute Phone Error Rate is not so important here and it can be certainly possible to devise recognisers with better performances by applying more sophisticated techniques. For example (Peddinti et al., 2015) presented a method for lattice rescoring in Kaldi that exhibits better performances than the n-best rescoring we used to interpolate between n-grams and the tested models, but modifying it in order to test LSTMLM and QLM presented a lot of problems and thus we decided to use the simpler n-best approach. For completeness, the last column of Table 2 outlines the results obtained using this lattice rescoring method with RNNLM as described in (Peddinti et al., 2015).

6 Discussion and conclusions

We presented a new technique for building LM based on QMT, and its probability calculus, testing it extensively both with intrinsic and extrinsic evaluation methods.

The PPL results for the intrinsic evaluation, outlined in Table 1, show a clear superiority of the proposed method when compared with state-of-the-art techniques such as RNNLM and LSTMLM. It is interesting to note that even using $D = 20$, that means a system containing a quarter of parameters, therefore much less 'memory', w.r.t. the system with $D = 40$, we obtain a PPL performance better than the other methods.

With regard to the second experiment we made, an extrinsic evaluation where we replaced the LM of an ASR system with the LM produced by all the tested methods (see Table 2), QLM consistently exhibits the best performances for all the tested ASR systems from the Kaldi "TIMIT recipe". Despite using a n-best technique in this evaluation for hypothesis rescoring, that is known to perform worse than the lattice rescoring method proposed in (Peddinti et al., 2015), the QLM performances are even better than this method.

The approach we have presented in this paper is not without problems: the number of different word types in the considered language has to be small in order to keep the model computationally tractable. Even if the code we used in the evaluations is analytically highly optimised, the training of this model is rather slow and requires relevant computational resources even for small problems. On the contrary, inference is very quick, faster than the RNNLM and LSTMLM packages we tested.

The main research question that drove this work was to verify if the distinguishing properties of quantum probability theory, namely interference and system entaglement that could allow the ancilla to have a "potentially infinite" memory, were enough to build stochastic systems more powerful than those built using classical probabilities or those built using recurrent NN. Our main aim was not to build a complete model to handle all possible LM scenarios, but to present a "proof-of-concept" study to test the potentialities of this approach. For this reason we tried to keep the model as simple as possible using orthogonal projectors: for measuring probabilities, projecting the system state, each word is mapped onto a single basis vector and the dimension of the system Hilbert space, N, is equal to the number of different words. Given the matrix dimensions that we have to manage when we add the ancilla, $DN \times DN$, this setting does not scale to real LM problems (e.g. the Brown corpus), even though the calculations are performed using $D \times D$ submatrices, but allowed us to successfully verify the research question. For the same reason out-of-vocabulary words cannot be handled in this model because there are no basis vectors assigned to them.

In order to overcome these limitations, this work can be extended by using generalized quantum measurements projectors (POVM) and by using a different structure for the system Hilbert space: instead of mapping each word onto a single basis vector we can span this space using as basis the same p-basis vectors used to define the V matrices. In this way we will project the system state on a generic word vector built as a superposi-

Kaldi ASR System	IRSTLM 2-gram	N-Best rescoring			Lattice rescoring
		IRSTLM 2-gram LM interp. with:			
		RNNLM	LSTMLM	QLM	RNNLM
tri1	26.32	25.74	*25.09*	**24.59**	25.70
tri2	24.14	23.34	23.23	**23.05**	*23.17*
tri3	21.55	21.07	21.22	**20.35**	*20.85*
SGMM2	19.15	18.99	*18.52*	**18.23**	18.75
Dan NN	22.27	22.20	22.26	**21.80**	*22.05*
Kaldi ASR System	**IRSTLM 3-gram**	**N-Best rescoring**			**Lattice rescoring**
		IRSTLM 3-gram LM interp. with:			
		RNNLM	**LSTMLM**	**QLM**	**RNNLM**
tri1	25.64	25.39	*24.86*	**24.59**	25.42
tri2	23.16	23.13	*22.90*	**22.65**	22.97
tri3	20.80	20.57	*20.68*	**20.04**	*20.68*
SGMM2	18.64	18.41	18.48	**18.23**	*18.27*
Dan NN	21.72	21.90	21.95	**21.34**	*21.48*

Table 2: Phone-recognition performances, in terms of Phone Error Rate, for the TIMIT dataset and the different Kaldi ASR models, rescoring the 10,000-best solutions with the tested LM techniques interpolated with the IRSTLM bigrams and trigrams LM (the standard LM used in Kaldi). In boldface the best performing system and in italics the second best. Kaldi ASR systems descriptions: tri1 = a triphone model using 13 dim. MFCC+Δ+$\Delta\Delta$; tri2 = tri1+LDA+MLLT; tri3 = tri2+SAT; SGMM2 = Semi-supervised Gaussian Mixture Model (Huang and Hasegawa-Johnson, 2010; Povey et al., 2011a); Dan NN = DNN model by (Zhang et al., 2014; Povey et al., 2015).

tion on the p-basis. Such improvement would reduce dramatically the dimensions of the matrices to $Dp \times Dp$ potentially mitigating the computational issue. Moreover, this would solve also the problem of out-of-vocabulary words allowing for a proper management of the large set of different words typical of real applications.

We are still working on these improvements and we will hope to get a complete model soon.

With this contribution we would like to raise also some interest in the community to analyse and develop more effective techniques, both on the modelling and minimisation/learning sides, to allow to build real world application based on this framework. QMT and its probability calculus seem to be promising methodologies to enhance the performances of our systems in NLP and certainly deserve further investigations.

Acknowledgments

We acknowledge the CINECA[4] award no. HP10C7XVUO under the ISCRA initiative, for the availability of HPC resources and support.

[4] https://www.cineca.it/en

References

Diederik Aerts, Jan Broekaert, Liane Gabora, and Sandro Sozzo. 2013. Quantum structure and human thought. *Behavioral and Brain Sciences*, 36(3):274276.

Martin Arjovsky, Amar Shah, and Yoshua Bengio. 2016. Unitary evolution recurrent neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - ICML'16*, pages 1120–1128.

Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. In Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade: Second Edition*, pages 437–478. Springer Berlin Heidelberg, Berlin, Heidelberg.

William Blacoe, Elham Kashefi, and Mirella Lapata. 2013. A quantum-theoretic approach to distributional semantics. In *Proceedings of Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Atlanta, Georgia*, pages 847–857.

Jerome R. Busemeyer and Peter D. Bruza. 2012. *Quantum Models of Cognition and Decision*. Cambridge University Press, New York, NY.

Philip Clarkson and Ronald Rosenfeld. 1997. Statistical language modeling using the cmu-cambridge

toolkit. In *Proceedings of EUROSPEECH '97*, pages 2707–2710. ISCA.

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. 2008. IRSTLM: an open source toolkit for handling large scale language models. In *INTERSPEECH 2008, 9th Annual Conference of the International Speech Communication Association, Brisbane, Australia*, pages 1618–1621.

John Garofolo, Lori Lamel, William Fisher, Jonathan Fiscus, David Pallett, Nancy Dahlgren, and Victor Zue. 1990. Darpa timit acoustic-phonetic continuous speech corpus cd-rom. *DARPA, TIMIT Acoustic-Phonetic Continuous Speech Corpus CD-ROM*.

Fabio A. González and Juan C. Caicedo. 2011. Quantum latent semantic analysis. In *Advances in Information Retrieval Theory, LNCS, 6931*, pages 52–63.

Jui Ting Huang and Mark Hasegawa-Johnson. 2010. Semi-supervised training of gaussian mixture models by conditional entropy minimization. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association, INTERSPEECH 2010*, pages 1353–1356.

Li Jing, Yichen Shen, Tena Dubcek, John Peurifoy, Scott A. Skirlo, Max Tegmark, and Marin Soljacic. 2017. Tunable efficient unitary neural networks (EUNN) and their application to RNN. In *Thirty-fourth International Conference on Machine Learning - ICML2017*.

Dimitrios Kartsaklis, Martha Lewis, and Laura Rimell. 2016. *Proceedings of the 2016 Workshop on Semantic Spaces at the Intersection of NLP, Physics and Cognitive Science*, volume 221. Electronic Proceedings in Theoretical Computer Science.

Andrei Y. Khrennikov. 2010. *Ubiquitous Quantum Structure: From Psychology to Finance*. Springer-Verlag Berlin Heidelberg.

Ding Liu, Xiaofang Yang, and Minghu Jiang. 2013. A novel classifier based on quantum computation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria*, pages 484–488.

Carla Lopes and Fernando Perdigao. 2011. Phoneme recognition on the timit database. In Ivo Ipsic, editor, *Speech Technologies*. InTech, Rijeka.

Massimo Melucci. 2015. *Introduction to Information Retrieval and Quantum Mechanics*. The Information Retrieval Series 35. Springer-Verlag Berlin Heidelberg.

Massimo Melucci and Keith van Rijsbergen. 2011. Quantum mechanics and information retrieval. In Massimo Melucci and Ricardo Baeza-Yates, editors, *Advanced Topics in Information Retrieval*, pages 125–155. Springer Berlin Heidelberg, Berlin, Heidelberg.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *Proc. of Workshop at ICLR*.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan*, pages 1045–1048.

Tomáš Mikolov, Stefan Kombrink, Anoop Deoras, Lukáš Burget, and Jan Černocký. 2011. Rnnlm - recurrent neural network language modeling toolkit. In *Proceedings of ASRU 2011*, pages 1–4.

Jorge J. Moré and David J. Thuente. 1994. Line search algorithms with guaranteed sufficient decrease. *ACM Trans. Math. Softw.*, 20(3):286–307.

Michael A. Nielsen and Isaac L. Chuang. 2010. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press.

J. Nocedal and S. J. Wright. 2006. *Numerical Optimization*, 2nd edition. Springer, New York.

Vijayaditya Peddinti, Guoguo Chen, Vimal Manohar, Tom Ko, Daniel Povey, and Sanjeev Khudanpur. 2015. JHU aspire system: Robust LVCSR with tdnns, ivector adaptation and RNN-LMS. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015, Scottsdale, AZ, USA*, pages 539–546.

Daniel Povey, Lukáš Burget, Mohit Agarwal, Pinar Akyazi, Feng Kai, Arnab Ghoshal, Ondřej Glembek, Nagendra Goel, Martin Karafiát, Ariya Rastrow, Richard C. Rose, Petr Schwarz, and Samuel Thomas. 2011a. The subspace gaussian mixture model-a structured model for speech recognition. *Comput. Speech Lang.*, 25(2):404–439.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. 2011b. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society.

Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur. 2015. Parallel training of dnns with natural gradient and parameter averaging. In *International Conference on Learning Representations - ICLR2015*.

Lutz Prechelt. 2012. Early stopping — but when? In Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade: Second Edition*, pages 53–67. Springer Berlin Heidelberg, Berlin, Heidelberg.

Shakti P. Rath, Daniel Povey, Karel Veselý, and Jan Cernocký. 2013. Improved feature processing for deep neural networks. In *Proceedings of the 14th Annual Conference of the International Speech Communication Association - INTERSPEECH2013, Lyon, France*, pages 109–113.

Daniel Soutner and Luděk Müller. 2015. In Adrian-Horia Dediu, Carlos Martín-Vide, and Klára Vicsi, editors, *Statistical Language and Speech Processing: Third International Conference, SLSP 2015, Budapest, Hungary, November 24-26, 2015, Proceedings*, pages 267–274. Springer International Publishing.

Christoph Spengler, Marcus Huber, and Beatrix C Hiesmayr. 2010. A composite parameterization of unitary groups, density matrices and subspaces. *Journal of Physics A: Mathematical and Theoretical*, 43(38):385306.

H.D. Tagare. 2011. Notes on optimization on Stiefel manifolds. Technical report, Technical report, Yale University.

Fabio Tamburini. 2014. Are quantum classifiers promising? In *Proceedings of the First Italian Conference on Computational Linguistics CLiC-it 2014, Pisa, Pisa University Press*, pages 360–364.

Vlatko Vedral. 2007. *Introduction to Quantum Information Science*. Oxford University Press, USA.

Scott Wisdom, Thomas Powers, John Hershey, Jonathan Le Roux, and Les Atlas. 2016. Full-capacity unitary recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 4880–4888. Curran Associates, Inc.

Xiaohui Zhang, Jan Trmal, Daniel Povey, and Sanjeev Khudanpur. 2014. Improving deep neural network acoustic models using generalized maxout networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 215–219. IEEE.

Guido Zuccon, Leif A. Azzopardi, and Keith van Rijsbergen. 2009. The quantum probability ranking principle for information retrieval. In Leif et al. Azzopardi, editor, *Advances in Information Retrieval Theory, LNCS, 5766*, pages 232–240.