Unitary Matrices As Compositional Word Embeddings

Jean-Philippe Bernardy Shalom Lappin

May 16, 2022

Theoretical linguistics is focussed on constructing theories that unify a variety of phenomena. A key characteristics of such theories is compositionality. The core of this property is the requirement that the content of a phrase is a function of the contents of its constituents. A weakness of many such theories is that they are brittle. That is, they break easily when applied at scale, because they do not cover the range of variation exhibited in real corpora. By contrast, statistical models, including recurrent neural networks (Hochreiter and Schmidhuber, 1997), do handle this variation, through Bayesian-style learning over a large parameter space. However, as these models improve in accuracy and coverage (Brown et al., 2020; Devlin et al., 2018; Solaiman et al., 2019; Yang et al., 2019), they become increasingly complex and less amenable to theoretical analysis. Thus, the gap between them and theoretical approaches expands.

In this abstract, we propose unitary-evolution recurrent neural networks (URNs) as a candidate to bridge this gap. URNs are both *compositional by design* and end-to-end trainable as statistical models. By the definition of unitary-evolution, at each step of the input sequence, a unitary transformation is applied to the state of the RNN. No activation function is applied between time-steps. Therefore, each input symbol can be interpreted as a unitary transformation, or equivalently as a unitary matrix. We can view the unitary matrices learned by a URN as *unitary word embeddings*. Because unitary matrices form a group, such embeddings can be composed and they constitute unitary embeddings for complex phrases.[1]

A consequence of compositionality is that the effect of embeddings can be analysed independently of context. Other statistical models (LSTMs, transformer) can only be analysed through black box testing methods, such as probing techniques (Hewitt and Manning, 2019). It is thus difficult to ascertain the syntactic structure that such models may encode.

Because unitary transformations are reversible, long distance dependency relations can, in principle, be reliably and efficiently recognised, without additional special-purpose machinery of the kind that (LSTM) RNNs require. This has been demonstrated to for copying and adding tasks (Arjovsky, Shah, and Bengio, 2016; Jing et al., 2017; Vorontsov et al., 2017). We show additionally that long-distance dependency effects are exhibited for several linguistically relevant syntactic tasks: (i) bracket matching in a generalised Dyck language, and (ii) the more challenging task of subject-verb number agreement in English.

[1]By constrast, the word embeddings that other deep neural networks learn are generally encoded as vectors. Combining such vector word embeddings into phrasal and sentence vectors is achieved through various *ad-hoc*, task-specific means, such as adding another trainable layer in the model. Each such a layer is a complicated, opaque model of its own. There is no compositionality as such.

URN architecture Since the simple recurrent networks of Elman (1990), the dominant architectures RNNs, including the influential LSTM Hochreiter and Schmidhuber, 1997, use non-linear activation functions ($sigmoid$, $tanh$, ReLU). By contrast our URNs invoke only linear cells. In fact, the cell that we use is a linear transformation of the unitary space, so that it takes unit state vectors to unit state vectors, hence the term "unitary-evolution".

For predictions, we extract a probability distribution from state vectors by applying a dense layer with softmax activation to each vector state s_i.

We need to ensure that $Q(x)$ is (and remains) orthogonal when it is subjected to gradient descent. In general, subtracting a gradient from an orthogonal matrix does not preserve orthogonality of the matrix. So we cannot make $Q(x)$ a simple lookup table from symbol to orthogonal matrix, without additional restrictions. While one could project the matrix onto an orthogonal space (Wisdom et al., 2016), our solution is to use a lookup table mapping each word to a skew-hermitian matrix $S(x)$. We follow Hyland and Rätsch (2017) in doing this. We then let $Q(x) = e^{S(x)}$, which ensures the orthogonality of $Q(x)$. It is not difficult to ensure that $S(x)$ is skew-symmetric. It suffices to store only the elements of $S(x)$ above the diagonal, and let those below it be their anti-symmetric image, with the diagonal set to zero.

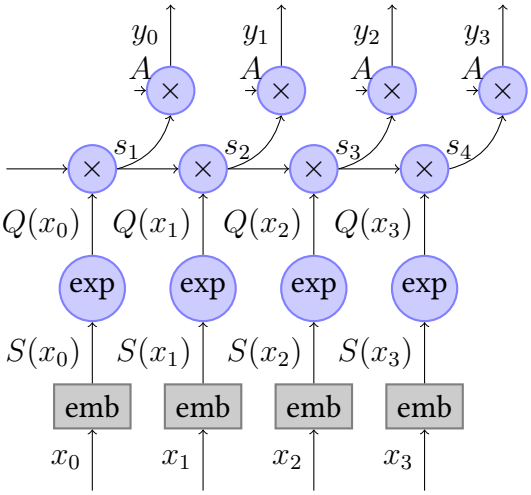

Figure 1: Each input symbol x_i indexes an embedding layer, yielding a skew-symmetric matrix $S(x_i)$. Taking its exponential yields an orthogonal matrix $Q(x_i)$. Multiplying the state s_i by $Q(x_i)$ yields the next state, s_{i+1}.

Experiments Our first experiment applies a URN to a natural language agreement task proposed by Linzen, Dupoux, and Golberg (2016). The model predicts the number of third person verbs in English text, with supervised training. In the phrase "The **keys** to the cabinet **are** on the table", the RNN is trained to predict the plural "**are**" rather than the singular "**is**".

Linzen, Dupoux, and Golberg (2016) point out that solving the agreement task requires knowledge of hierarchical syntactic structure. If an RNN captures the long-distance dependencies involved in agreement relations, it cannot rely solely on the linear sequence of nouns (in particular their number inflections) preceding the predicted verb in a sentence. Accuracy must be sustained as the number of *attractors* increases. An attractor is defined as a noun occurring between the subject and the verb, which has the wrong number feature for controlling the verb. In the above example, "cabinet" is an attractor.

The second experiment evaluates the long-distance modelling capabilities of an RNN in a way that abstracts away from the noise in natural language, by using synthetic data. Following Bernardy (2018) we employ a (generalised) Dyck language. It is composed solely of matching parenthesis pairs. So the strings "`{([])}<>`" and "`{()[<>]}`" are part of the language, while "`[}`" is not. This experiment is an idealised version of the agreement task, where opening parentheses correspond to subjects, and closing parentheses to verbs. An attractor is an opening parenthesis occurring between the pair, but of a different kind. Matching of parentheses corresponds to agreement.

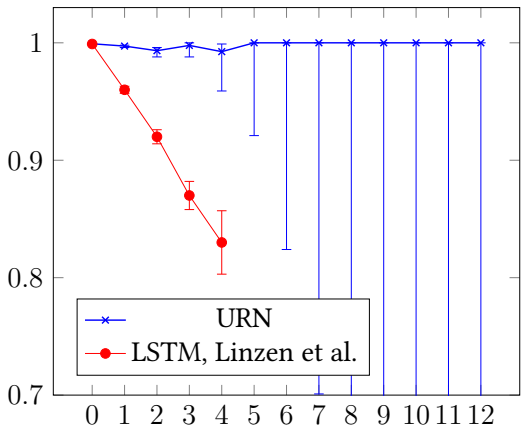

Figure 2: Accuracy per number of attractors for the verb number agreement task. Linzen, Dupoux, and Golberg (2016) do not report performance of their LSTM past 4 attractors. Error bars represent binomial 95% confidence intervals. We see that the URN "solves" this task, with error rates well under one percent. Crucially, there is no evidence of accuracy dropping as the number of attractors increases. Even though the statistical uncertainty increases with the number of attractors, due to decreasing numbers of examples, the URN makes no mistakes for higher number of attractors.

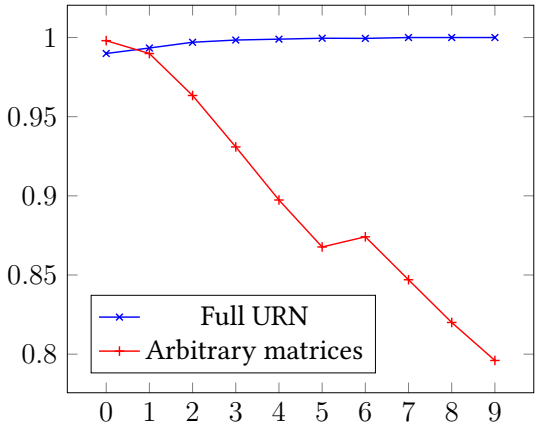

Figure 3: Accuracy of closing parenthesis prediction by number of attractors. The size of matrices is 50 by 50.

References

Arjovsky, Martin, Amar Shah, and Yoshua Bengio (2016). "Unitary Evolution Recurrent Neural Networks". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA: JMLR.org, pp. 1120–1128.

Bernardy, Jean-Philippe (2018). "Can RNNs Learn Nested Recursion?" In: *Linguistic Issues in Language Technology* 16 (1).

Brown, T. et al. (2020). "Language Models are Few-Shot Learners". In: *ArXiv* abs/2005.14165.

Devlin, Jacob et al. (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805*.

Elman, Jeffrey L. (1990). "Finding structure in time". In: *Cognitive Science* 14.2, pp. 179–211.

Hewitt, John and Christopher D. Manning (June 2019). "A Structural Probe for Finding Syntax in Word Representations". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4129–4138.

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997). "Long short-term memory". In: *Neural Computation* 9.8, pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735.

Hyland, Stephanie L and Gunnar Rätsch (2017). "Learning unitary operators with help from u (n)". In: *Thirty-First AAAI Conference on Artificial Intelligence*.

Jing, Li et al. (2017). "Tunable Efficient Unitary Neural Networks (EUNN) and their application to RNN". In: *arXiv*.

Linzen, Tal, Emmanuel Dupoux, and Yoav Golberg (2016). "Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies". In: *Transactions of the Association of Computational Linguistics* 4, pp. 521–535.

Solaiman, Irene et al. (2019). "Release Strategies and the Social Impacts of Language Models". In: *ArXiv* abs/1908.09203.

Vorontsov, Eugene et al. (2017). "On Orthogonality and Learning Recurrent Networks with Long Term Dependencies". In: *arXiv*.

Wisdom, Scott et al. (2016). "Full-capacity unitary recurrent neural networks". In: *Advances in neural information processing systems* 29, pp. 4880–4888.

Yang, Zhilin et al. (2019). "XLNet: Generalized Autoregressive Pretraining for Language Understanding". In: *ArXiv* abs/1906.08237.