Serial Recall Effects in Neural Language Modeling

Hassan Hajipoor[1], **Hadi Amiri**[2], **Maseud Rahgozar**[1], **Farhad Oroumchian**[3]
[1]School of Electrical and Computer Engineering, University of Tehran, Iran
[2]Department of Biomedical Informatics, Harvard University, Boston, USA
[3]University of Wollongong in Dubai, Knowledge Village, Dubai, UAE
h.hajipoor@ut.ac.ir, hadi@hms.harvard.edu,
rahgozar@ut.ac.ir,farhadoroumchian@uowdubai.ac.ae

Abstract

Serial recall experiments study the ability of humans to recall words in the order in which they occurred. The following serial recall effects are generally investigated in studies with humans: *word length* and *frequency*, *primacy* and *recency*, *semantic confusion*, *repetition*, and *transposition* effects. In this research, we investigate LSTM language models in the context of these serial recall effects. Our work provides a framework to better understand and analyze neural language models and opens a new window to develop accurate language models.

1 Introduction

The goal of language modeling is to estimate the probability of a sequence of words in natural language, typically allowing one to make probabilistic predictions of the next word given preceding ones (Bahl et al., 1983; Berger et al., 1996). For several years now, Long Short-Term Memory (LSTM) (Graves, 2013) language models have demonstrated state-of-the-art performance (Melis et al., 2018; Merity et al., 2018a; Sundermeyer et al., 2012). Recent studies have begun to shed light on the information encoded by LSTM networks. These models can effectively use distant history (about 200 tokens of context) and are sensitive to word order, replacement, or removal (Khandelwal et al., 2018), can learn function words much better than content words (Ford et al., 2018), can remember sentence lengths, word identity, and word order (Adi et al., 2017), can capture syntactic structures (Kuncoro et al., 2018) such as subject-verb agreement (Linzen et al., 2016). These characteristics are often attributed to LSTM's ability in overcoming the *curse of dimensionality*–by associating a distributed feature vector to each word (Hinton et al., 1986; Neubig, 2017)–and modeling long-range dependencies in faraway context (Khandelwal et al., 2018).

The goal of our research is to complement the prior work to provide a richer understanding about how LSTM language models use prior linguistic context. Inspired by investigations in cognitive psychology about serial recall in humans (Avons et al., 1994; Henson, 1998; Polišenská et al., 2015)–where participants are asked to recall a sequence of items in order in which they were presented, we investigate how word length or frequency (*word-frequency* effect), word position (*primacy*, *recency*, and *transposition* effects), word similarity (*semantic confusion* effect), and word repetition (*repetition* effect) influence learning in LSTM language models. Our investigation provides a framework to better understand and analyze language models at a considerably finer-grained level than previous studies, and opens a new window to develop more accurate language models.

We find that LSTM language models (a) can learn frequent/shorter words considerably better than infrequent/longer ones, (b) can learn recent words in sequences better than words in earlier positions,[1] (c) have a tendency to predict words that are semantically similar to target words - indicating that these networks have a tendency to group semantically similar words while suggesting one specific word as target based on prior context, (d) predict as output the words that are observed in prior context, i.e. repeat words from prior context, and (e) may transpose (switch adjacent) words in output depending on word syntactic function.

[1]Rats and humans recall the first and last items of sequences best and the middle ones worst (Bolhuis and Van Kampen, 1988; Ebbinghaus, 1913).

Proceedings of NAACL-HLT 2019, pages 688–694
Minneapolis, Minnesota, June 2 - June 7, 2019.

2 Serial Recall Effects

Language models estimate the probability of a sequence as $P(w_1^n) = \prod_{i=1}^{n} P(w_i|w_1^{i-1})$, where w_i is the ith word and w_i^j indicates the sub-sequence from w_i to w_j. These models minimize their prediction error against words in context, using e.g. the negative log likelihood loss function, during training: $\mathcal{L} = -\frac{1}{n}\Sigma_{i=1}^{n} \log P(w_i|w_1^{i-1})$. In this paper, we show the loss of a model against sequence s and word $w_i \in s$ by $\mathcal{L}(s)$ and $\mathcal{L}(s)[w_i]$ respectively.

We use the LSTM language model developed in (Merity et al., 2018b) for our experiments. Given a sequence of words w_1^{i-1} as context, this model predicts the next word in sequence. We refer to w_i and $\hat{w}_i$ as *target* and *predicted* words respectively; given the global vocabulary $\mathcal{V}$, $\hat{w}_i = \arg\max_{w_j \in \mathcal{V}} \Pr(w_j|w_1^{i-1})$. We study this LSTM in the context of serial recall effects.

2.1 Word-Frequency Effect

What is the effect of word frequency/length[2] on the performance of LSTM language models? For this effect, we report the average loss for each word frequency as follows:

$$\mathcal{L}_{WF}^{k} = 1/|\mathcal{S}_k| \sum_{s \in \mathcal{S}_k, w_i \in s} \mathcal{L}(s)[w_i], \qquad (1)$$

where $\mathcal{S}_k$ is the set of sequences that, at least, have one target word with term frequency of k, and $\mathcal{L}_{WF}^{k}$ is the overall loss for target words of frequency k which sheds light on the expected frequency of words for accurate language modeling.

2.2 Primacy and Recency Effect

What is the effect of word position on the performance of LSTM language models? To analyze this effect, we compute the average loss of network with respect to the position of target words as follows:

$$\mathcal{L}_{PR}^{i} = 1/Z \sum_{s} \mathcal{L}(s)[w_i], \qquad (2)$$

where w_i is the target word in sequence s, Z is the number of sequences as normalization factor, and $\mathcal{L}_{PR}^{i}$ is the average of loss at position i. This effect will shed light on network performance at specific positions in texts which can help rationalizing the need for new language modeling architectures, such as bidirectional LSTMs (Schuster and Paliwal, 1997), or the order by which input data should be processed (Sutskever et al., 2014).

[2]Note that word length and frequency are directly correlated (Bell et al., 2009).

2.3 Semantic Confusion Effect

Are predicted words semantically similar to the target ones in case of incorrect predictions? For this analysis, we report the average semantic similarity between target (w_i) and predicted ($\hat{w}_i$) words as follows:

$$\mathcal{SC} = 1/Z \sum_{s, w_i \in s, w_i \neq \hat{w}_i} \texttt{sim}(w_i, \hat{w}_i), \qquad (3)$$

where the function $\texttt{sim}(.,.)$ computes word similarity either through WordNet (Miller, 1998) or `cosine` similarity of the corresponding embeddings of its arguments. This effect will shed light on how effective LSTMs are in disentangling semantically-similar concepts. This gives us a powerful metric to compare networks semantically, especially, in case of equal loss/perplexity.

2.4 Repetition Effect

This effect refers to prediction of a word that already exists in context, i.e. in an earlier position in the sequence. Here, for each target w_n, we compute the probability that, instead of w_n, any of w_1^{n-1} words is predicted as output and report average across all samples:

$$\Pr(RE^i) = \qquad (4)$$
$$1/Z \sum_{s} \Pr(w_i \in w_1^{n-1}, w_i \neq w_n | w_1^{n-1}).$$

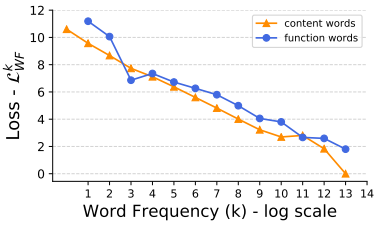
This effect will shed light on the extent to which network repeats/predicts observed words as possible responses. Given that words rarely repeat in sentences, the above metric can be used as a good regularizer for language modeling.

2.5 Transposition Effect

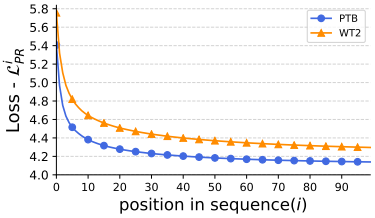
This effect refers to word prediction in transposed positions, i.e. the case where the word pair w_i^{i+1} in an original sequence is more likely to be predicted by the network as $w_{i+1}w_i$ in output. Here, for each pair w_i^{i+1} in target sequence, we count the number of times in which w_{i+1} is more probable to be predicted at position i (as compared to w_i) and w_i is more probable to be predicted at position $i+1$ (as compared to w_{i+1}). We report the average number of transposition occurrences for all samples at each word position. This effect will shed light on how network learn nearby grammatical orders such as conjunction and adjective order.

	PTB		WT2	
	#tokens	#words	#tokens	#words
NN	21k	3k	55k	8k
VB	10k	1.7k	25k	3k
RB	2.6k	286	5k	0.5k
JJ	4.9k	0.9k	12k	1.8k
Func	34.9k	0.2k	118k	0.9k

(a) Dataset statistics



(b) Word frequency vs. loss (PTB)



(c) Primacy and Recency effect

Figure 1: (a) Dataset statistics, (b): Word-frequency effect, network learns frequent words better than infrequent ones, (c): Primacy and Recency effect, loss decreases for target words that appear at the end of sequences.

3 Experiments

Datasets. We use two benchmark language modeling datasets: Penn Treebank (PTB) (Marcus et al., 1993; Mikolov et al., 2010) and WikiText-2 (WT2) (Merity et al., 2017). PTB and WT2 have vocabulary sizes of 10K and 33K respectively. We use the POS-tagged versions of these datasets provided by Khandelwal et al. (2018), and treat nouns (NN), verbs (VB), adjectives (JJ), and adverbs (RB) as content words, and others word classes as function words, see details in Figure 1a.

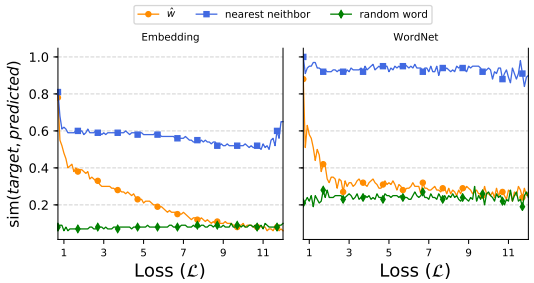
Settings. We set LSTM's parameters as suggested in (Merity et al., 2018b) for PTB and follow its suggested parameter tuning procedure for WT2. For both datasets, we set context size to $n = 100$ obtained from $\{5, 20, 50, 100, 200\}$ and validation data; note that the number of samples are equal for different sequence lengths.

3.1 Results

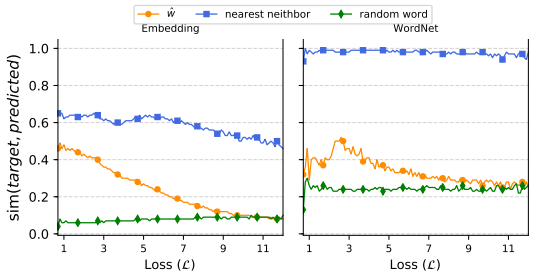
We report LSTM performance in terms of prediction loss on development sets for all experiments.

Word-Frequency Effect: *More frequent target words are predicted (learned) more accurately than less frequent ones.* Figure 1b shows strong inverse correlation between word frequency and LSTM prediction loss. This is expected as neural models learn better with more data. In addition, although the overall loss of function words is considerably lower than that of content words (because of their overall higher frequency), Figure 1b shows that, for the same word frequency, content words are learned better than function words.

Primacy and Recency Effects: *Target words that appear later in sequences are predicted considerably better than those at earlier positions.* Figure 1c shows that prediction loss considerably decreases for target words that appear toward the end of the sequences. The results are consistent across



(a) Semantic confusion effect in PTB.

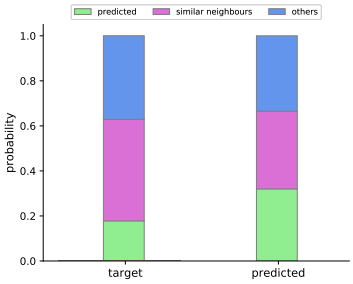


(b) Semantic confusion effect in WT2.

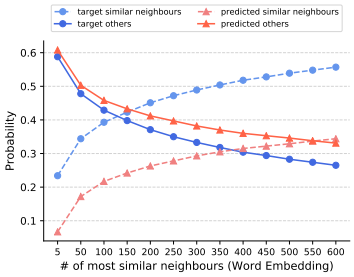
Figure 2: We report semantic similarity between predicted and target words, random and target words (lower bound), and nearest neighbor and target words (upper bound).

both datasets. This effect can explain why bidirectional LSTMs which read input from opposite directions usually work better in NLP applications such as machine translation (Firat et al., 2016; Domhan, 2018). A remaining question that is worth investigating is whether bidirectional LSTMs learn first and last few words of sequences better than those in the middle, and if yes, how can we make these models more robust against word position.

Semantic Confusion Effect. *There is significant tendency to predict words that resemble (are semantically-similar to) target words.* Figure 2 shows the average WordNet and Embedding similarity between target and predicted words in PTB and WT2 across loss values. The results indicate high similarity between predicted and target words

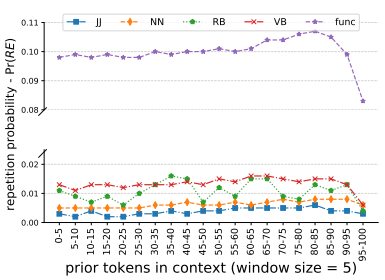


(a) Proportion of neighbors from prediction probability (1 − Loss)

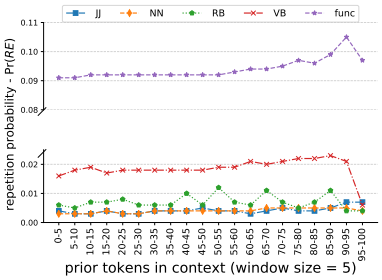


(b) Finding appropriate neighbors size

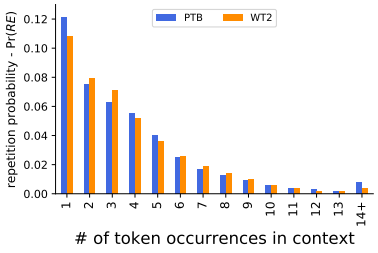
Figure 3: Semantic confusion effect. (a) The chance of neighbors of target is equal to chance of other words. (b) When the neighbours size is set to 150, the probability of target neighbouring group became equal to the probability of target. Neighbours size of 600, leads to equal chance for predicted token.



(a) Repetition effect in PTB



(b) Repetition effect in WT2



(c) Repetition vs. occurrence in context

Figure 4: (a, b): Repetition effect across POS tag classes in PTB and WT2 and (c): Repetition probability decreases as a function of word occurrences in prior context.

for smaller loss values. However, confusion consistently increases as prediction loss increases. The upper bound similarity (obtained by treating the nearest neighbor of each target word as predicted word) indicates there exists better candidates which LSTM fails to predict. Our further analyses show that LSTM has a tendency to group semantically similar words and then suggest one of them. We consider most similar words as a group of neighbors and examine how network assigns probabilities to them as compared to others. Figure 3a shows that the chance of neighbors of target (with size 150) is equal to chance of other words (with size 9849). As Figure 3a shows, the neighbors of predicted words (with size 600) carry equal chance as compared to other words (with size 9399). To find these thresholds, we gradually increase the number of neighbors and track the trend of approaching the probability of neighboring group to target. As shown in Figure 3b, if the size of neighboring group is set to 150, these probabilities became equal. The similar way is repeated for finding the appropriate size for neighbors of predicted words.

Repetition Effect. *Repetition probability of function words is significantly higher than that of content words*. We report repetition effect, see Eq. (4), at POS tag level (where the predicted word should have the same POS tag as target word in prior context). As Figure 4a and 4b show, function words have higher repetition probability than content words. This is because function words are more frequent, and the average distance among them (i.e. number of intervening words) is considerably smaller than other POS tags (e.g. 2.1 words vs 28.9 and 12.4 words for RB and JJ respectively). We also find that repetition probability decreases as a function of word frequency in prior context, see Figure 4c. This is because words (especially NNs and VBs) are often self-contained and their occurrence in prior context helps LSTM to avoid "repeating" them. In addition, we find that function words repeat more frequently than other types and repetition among NNs and VBs is higher than other POS tag pairs; Table 1 shows the confusion matrix for repetition across POS tag classes Perhaps, this could explain the recent language modeling improvement obtained in (Ford et al., 2018) through developing separate yet sequentially connected network architectures with respect to POS tag class.

There are two factors determining the chance of

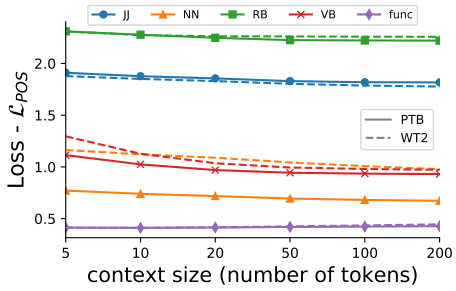


Figure 5: Effective context to learn syntax.

	NN	VB	JJ	RB	Func
NN	**0.07**	0.00	0.01	0.00	**0.24**
VB	0.01	**0.09**	0.00	0.01	**0.20**
JJ	0.03	0.01	0.01	0.00	**0.27**
RB	0.01	0.02	0.00	0.03	**0.31**
Func	0.01	0.01	0.00	0.00	**0.52**

Table 1: Confusion matrix for repetition effect. Rows and columns show POS tag classes of target and predicted words respectively.

repetition of a POS class: First, the average distance of consecutive tokens of that POS class; Table 2 reports the corresponding values from training set. Second, the accuracy of network in predicting POS classes which has been shown in Figure 5 and also reported in (Khandelwal et al., 2018). From these, the repetition probability of function words are expected to be higher than content words.

	NN	VB	JJ	RB	Func
PTB	3.6	7.1	12.4	28.9	2.1
WT2	3.6	8.5	15.8	38.5	1.9

Table 2: Average distance of tokens in POS classes.

Transposition Effect: *Transpositions occur more frequently at the beginning of sequences and rarely at the end*. Figure 6 shows average number of transpositions at each word position across datasets. This result is meaningful because miss-predictions occur more frequently at earlier positions (see results for primacy and recency effect). In addition, transpositions are rare at higher positions because more context information can help LSTM to make accurate (transposition-free) prediction. In addition, Table 3 shows the percentage of transpositions across POS tag classes on PTB. The result show that LSTM mainly transposes '`RB NN`' word pairs with '`NN RB`.' In future, we will conduct further analyses to understand the reason.

The findings presented in this paper provide insight into how LSTMs model context. This information can be useful for improving language models. For instance, the discovery that some word types are repeated in predictions more than others can help regularizing neural language models by making them adaptive to the different word types.

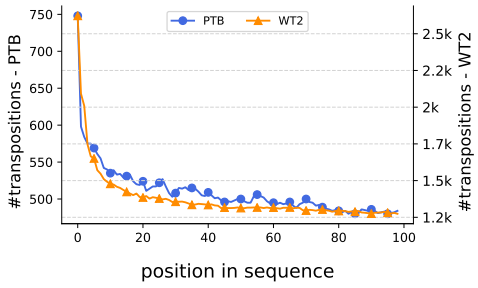


Figure 6: Transposition effect, transpositions occur more at the beginning of sequences.

	NN	VB	JJ	RB	Func
NN	0.025%	0.006%	0.032%	0.015%	0.001%
VB	0.011%	0.001%	0.004%	0.002%	0.001%
JJ	0.021%	**0.054%**	**0.045%**	**0.036%**	0.002%
RB	**0.073%**	0.017%	0.017%	0.023%	0.012%
Func	0.002%	0.002%	0.002%	0.001%	0.006%

Table 3: Confusion matrix for transpositions in PTB at POS tag level. LSTM transposes '`RB NN`' and '`JJ VB|JJ|RB`,' more than others pairs.

4 Conclusion

We investigate LSTM language models in the context of serial recall indicators. We find that frequent target words and target words that appear later in sequences are predicted more accurately, predictions often resemble (are semantically similar to) target words, function words of prior context are more likely to be predicted as target words, and word pair transpositions occur more frequently at the beginning of sequences.

Acknowledgments

We sincerely thank Nazanin Dehghani for her constructive collaboration during development of this paper and anonymous reviewers for their insightful comments and effective feedback.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *International Conference on Learning Representations (ICLR)*.

SE Avons, KL Wright, and Kristen Pammer. 1994. The word-length effect in probed and serial recall. *The*

Quarterly Journal of Experimental Psychology Section A, 47(1):207–231.

Lalit R Bahl, Frederick Jelinek, and Robert L Mercer. 1983. A maximum likelihood approach to continuous speech recognition. *IEEE transactions on pattern analysis and machine intelligence*, (2):179–190.

Alan Bell, Jason M Brenier, Michelle Gregory, Cynthia Girand, and Dan Jurafsky. 2009. Predictability effects on durations of content and function words in conversational english. *Journal of Memory and Language*, 60(1):92–111.

Adam L Berger, Vincent J Della Pietra, and Stephen A Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

Johan J Bolhuis and Hendrik S Van Kampen. 1988. Serial position curves in spatial memory of rats: Primacy and recency effects. *The Quarterly Journal of Experimental Psychology*, 40(2):135–149.

Tobias Domhan. 2018. How much attention do you need? a granular analysis of neural machine translation architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1799–1808.

H Ebbinghaus. 1913. Memory: A contribution to experimental psychology (ha ruger & ce bussenius, trans.). new york, ny, us.

Orhan Firat, Kyunghyun Cho, and Yoshua Bengio. 2016. Multi-way, multilingual neural machine translation with a shared attention mechanism. In *Proceedings of NAACL-HLT*, pages 866–875.

Nicolas Ford, Daniel Duckworth, Mohammad Norouzi, and George Dahl. 2018. The importance of generation order in language modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2942–2946.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Richard NA Henson. 1998. Short-term memory for serial order: The start-end model. *Cognitive psychology*, 36(2):73–137.

Geoffrey E Hinton et al. 1986. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, volume 1, page 12. Amherst, MA.

Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. 2018. Sharp nearby, fuzzy far away: How neural language models use context. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 284–294. Association for Computational Linguistics.

Adhiguna Kuncoro, Chris Dyer, John Hale, Dani Yogatama, Stephen Clark, and Phil Blunsom. 2018. Lstms can learn syntax-sensitive dependencies well, but modeling structure makes them better. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1426–1436.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.

Gábor Melis, Chris Dyer, and Phil Blunsom. 2018. On the state of the art of evaluation in neural language models. *International Conference on Learning Representations*.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018a. An analysis of neural language modeling at multiple scales. *arXiv preprint arXiv:1803.08240*.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018b. Regularizing and optimizing lstm language models. In *International Conference on Learning Representations*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *International Conference on Learning Representations*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

George Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

Graham Neubig. 2017. Neural machine translation and sequence-to-sequence models: A tutorial. *arXiv preprint arXiv:1703.01619*.

Kamila Polišenská, Shula Chiat, and Penny Roy. 2015. Sentence repetition: what does the task measure? *International Journal of Language and Communication Disorders*, 50(1):106–118.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Martin Sundermeyer, Ralf Schlúter, and Hermann Ney. 2012. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.