2020

Mutual Information Decay Curves and Hyper-parameter Grid Search Design for Recurrent Neural Architectures

Abhijit Mahalunkar
Technological University Dublin, abhijit.mahalunkar@tudublin.ie

John Kelleher
Technological University Dublin, john.kelleher@tudublin.ie

Recommended Citation

Mahalunkar, A & Kelleher, John D. (2020) Mutual Information Decay Curves and Hyper-parameter Grid Search Design for Recurrent Neural Architectures,*In book: Neural Information Processing, 27th International Conference, ICONIP 2020, Bangkok, Thailand, November 18–22, 2020, Proceedings, Part V (pp.616-624) DOI:10.1007/978-3-030-63823-8_70*

Mutual Information Decay Curves and Hyper-Parameter Grid Search Design for Recurrent Neural Architectures

Abhijit Mahalunkar[0000−0001−5795−8728] and
John D. Kelleher[0000−0001−6462−3248]

ADAPT Research Center, Technological University Dublin, Ireland
{abhijit.mahalunkar,john.d.kelleher}@tudublin.ie

Abstract. We present an approach to design the grid searches for hyper-parameter optimization for recurrent neural architectures. The basis for this approach is the use of mutual information to analyze long distance dependencies (LDDs) within a dataset. We also report a set of experiments that demonstrate how using this approach, we obtain state-of-the-art results for DilatedRNNs across a range of benchmark datasets.

Keywords: Long Distance Dependencies · Recurrent Neural Architectures · Hyper-Parameter Tuning · Vanishing Gradients.

1 Introduction

Recurrent neural networks trained using backpropagation through time suffer from exploding or vanishing gradients [7,8,1,10]. This problem presents a specific challenge in modeling sequential datasets which exhibit long distance dependencies (LDDs) [13]. LDDs describe an interaction between two (or more) elements in a sequence that are separated by an arbitrary number of positions. This results in the decay of statistical dependence of two points with increasing distance between them. Building recurrent neural architectures that are able to model LDDs is an open research problem and much of the current research on the topic focuses on designing new neural architectures. Early work in this direction proposed a hierarchical recurrent neural network that introduced several levels of state variables, working at different time scales [5]. This work inspired other architectures, such as DilatedRNN [3], Skip RNN [2], etc. Other well-known approaches to address this challenge are [9,14,6,16,17,4]. In this paper, we argue that a key step in designing recurrent neural architectures is to understand the decay of dependence in sequential data and to use this understanding to inform the setting of the relevant hyper-parameters of the architecture. In previous work, we developed an algorithm to compute and visualize the decay of dependence of the symbols within the dataset [12]. In this paper, we build on this previous work and show how this type of analysis can inform the selection of hyper-parameters of existing recurrent neural architectures. In this regard, we use DilatedRNNs [3] as a test model, and for several benchmark datasets, we study the decay of

dependence of a dataset and then design a set of dilations tailored to that dataset. With this approach, we achieve better performance as compared to the technique mentioned in the original implementation [3].

2 Dilated Recurrent Neural Networks

In this paper, we demonstrate how the analysis of the decay of dependence is a useful source of information to guide the selection of hyper-parameters to model the sequential data. We have chosen to DilatedRNNs [3] as test model for our experiments because of the relatively transparent relationship between some of the hyper-parameters of this architecture and the ability of the network to model LDDs. The DilatedRNN architecture is a multi-layer and cell-independent architecture characterized by multi-resolution dilated recurrent skip-connections. This alleviates the gradient problems and extends the range of temporal dependencies. Upon stacking multiple dilated recurrent layers with increasing skip-connections, these networks can learn temporal dependencies at different scales. DilatedRNN architecture and the dilations are described in [3]. The size of dilations per layer and the number of layers are supplied using the *dilations* hyper-parameter. This hyper-parameter controls the gradient flow and memory capacity of the DilatedRNNs. We hypothesize that to optimize the performance of a DilatedRNN on a dataset, the set of dilations should be tailored to match LDDs within the dataset. In particular: 1) the max dilation should match the span of LDDs present in the dataset (i.e. the dilation length should not extend to distances where there is low mutual information), 2) the increase in the size of dilation per layer should match the decay of dependence observed in the dataset (i.e. in regions where there is a rapid decay of dependence, there should be dense set of skip-connections capturing the dependence).

The process we propose for fitting the dilations hyper-parameter of these networks to a dataset is as follows: 1) plot the dependency decay curve [12], 2) design a grid search over the hyper-parameter space using the framework presented in this paper, 3) fit the model to the data and evaluate. To demonstrate this approach, we train DilatedRNNs on MNIST and Penn Treebank (PTB).

3 Interpreting Dependency Decay Curve to Inform Hyper-Parameter Optimization

In [12], we proposed an algorithm to analyze the decay of dependence in sequential datasets. This analysis can be visualised by plotting this decay on a log-log axis, where the x-axis describes the distance d between two observations in the sequence and the y-axis describes the mutual information (MI) at that d (in *nats*). We call these plots dependency decay curves and this paper focuses on how the analysis of these plots can be used to guide hyper-parameter selection. The decay of dependence either follows exponential decay, indicating the absence of LDDs, or power-law decay, indicating the presence of LDDs. Furthermore, the influence

of various phenomena on LDDs can lead to decay curve following: 1) power-law decay, 2) power-law decay with a periodicity present, or 3) broken power-law. A broken power-law is made up of multiple power-laws joined at inflection point. Fig. 1 illustrates how these different types of decay are exhibited in an dependency decay curve plot. As dependency decay curves are plotted on a log-log axis a curve that follows a straight line represents a power-law decay (e.g. Fig. 1(a)).

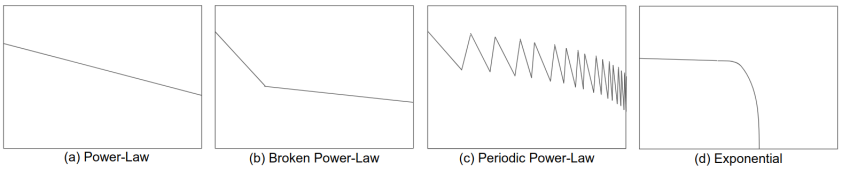


Fig. 1. Dependency decay curve framework which displays standard LDD curves

In the context of DilatedRNNs the best dilation hyper-parameters to model a dataset exhibiting power-law decay are: 1) max dilation should be equal to d where the dependency decay curve crosses MI=10^{-5} (this threshold is informed by [11] as MI below this is assumed to be noise); and 2) the set of dilations should be designed such that the density of skip-connections in a region should increase when the slope is high and decrease when it is low. The standard progression of dilations used for DilatedRNNs is $1, 2, 4, 8, \ldots$ and these dilations are a good guide for reasonable rates of dependency decay (similar to the slope in Fig. 1(a)). However, if the line gets steeper the spacing between the skip-connections should become smaller and denser e.g. $1, 2, 3, 4, 5, \ldots$. Fig. 2 illustrates the design pattern we use to connect the slope of the decay of dependence to the density of the skip-connections in a DilatedRNN. In Fig. 2(a) and 2(b), we see two power-law decays in two plots. In each plot we have drawn a sequence of equidistant horizontal lines representing hidden layers. We use the density of the x-intercept of every horizontal line to inform the patterning of the skip-connections for every layer in the network.

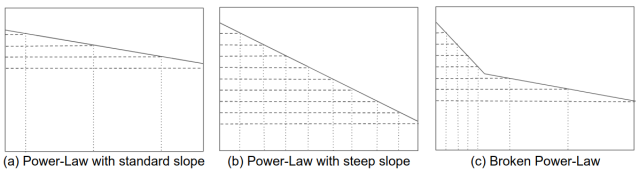


Fig. 2. Mapping set of dilations to dependency decay curve

Fig. 1(b) is an dependency decay curve plot that exhibits a broken power-law [15]. The hyper-parameter selection for such plots follow the same rules as

that of the power-law decay curve, with one major difference being that the presence of multiple power-laws will require different patterning of the dilations in different segments of the broken power-law in order to model the different rates of MI decay. Fig. 2(c), shows that when the steeper power-law changes to a shallower power-law, the dilations become more sparse. Fig. 1(c) is a dependency decay curve plot that is exhibiting a power-law decay with periodicity. The presence of periodicity within the decay indicates that the max dilation parameter should be set to the period of the MI peaks. The set of dilations can still follow the standard pattern for DilatedRNNs. An exponential decay involves a rapid decay in mutual information, such as that shown in Fig. 1(d). Such a rapid decay indicates the absence of LDDs. So in order to model such datasets, the max dilation should be set to where the dependency decay curve crosses the MI=10^{-5}. It can follow the standard set of dilations of the DilatedRNNs.

4 Dependency Decay Curves of Benchmark Datasets

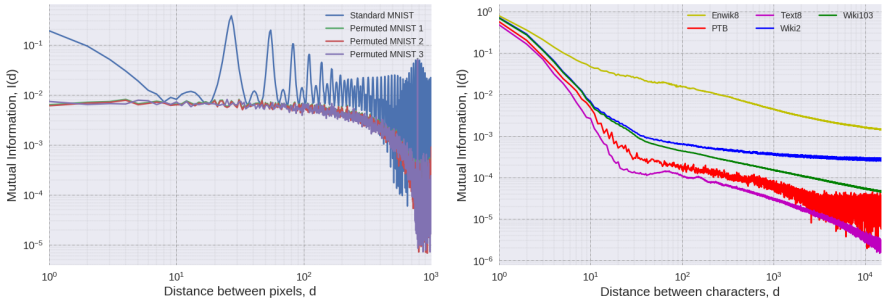
4.1 Sequential MNIST

Sequential MNIST is widely used to evaluate recurrent neural architectures. It contains 240000 training and 40000 test images. Each of these is 28x28 pixels in size, and each pixel can take one of 256 pixel values. In order to use them in a sequential task, the 2D images are converted into a 1D vector of 784 pixels by concatenating all the rows of the pixels. This transformation results in pixel dependencies which span up to approximately 28 pixels. These dependencies arise due to high correlation of a pixel with its neighboring pixels. The structure of the Sequential MNIST dataset is such that its dependency decay curve is likely to contain regular peaks and troughs. We plot the dependency decay curve of the unpermuted and permuted sequential MNIST datasets in Fig. 3(a).

Standard sequential MNIST exhibits high MI at D=1 indicating strong dependencies at close proximity. The dependencies then decay as a function of power-law. Hence, in-order to fully capture these dependencies, the recurrent neural architectures should maintain gradients/attention across multiple timescales as a function of power-law to accurately model these dependencies. However, we see peaks of MI at intervals of 28 due to pixel dependencies. The regular peaks in the decay curve indicate that the span of the dependencies lie within $D\approx$28.

We generated *permuted* versions of the sequential MNIST dataset with multiple seeds for use as a comparator with the unpermuted sequential MNIST. When we examine the dependency decay curves of permuted MNIST datasets, we observe that the dependencies are substantially less between close-by symbols (pixels in this case), e.g. for D=1 the green, red and purple lines are much lower than the blue line. This is a result of permutations applied to the data which disrupt spatial dependencies. Another impact of this disruption is the relatively flat curve for D<300 which indicates an absence of spatial dependencies. In-order to model these datasets that exhibit a relatively flat curve, the recurrent neural architectures requires uniform distribution of attention/gradients across all time scales. However, beyond D>300, we observe exponential decay of dependence,

where the value of MI falls below 10^{-5} indicating no further dependencies. This point ($D \approx 780$) indicates the span of the dependencies and the limit on the memory capacity of recurrent neural architectures.



(a) Unpermuted & Permuted sequential MNIST (multiple seeds)

(b) Character-based language

Fig. 3. Dependency Decay Curves

4.2 Character-Based Datasets

There are a number of natural language datasets that are frequently used to benchmark recurrent neural architectures, such as the Penn TreeBank (PTB), Wiki2, Wiki103, Text8 and Enwik8. These datasets can either be analysed at the word or character level. In this section we focus our analysis at the character level. Fig. 3(b) shows the dependency decay curve of these character-based datasets. These plots follow a power-law decay function. Except Enwik8, which follows a single power-law decay, all the other datasets (PTB, Wiki2, Wiki103 and Text8) follow multiple *power-laws* with an inflection point[1]. The character-based datasets exhibit two distinct decay curves i) single power-law decay (Enwik8), and ii) broken power-law decay (the rest of the curves). The optimal hyper-parameters are selected based on the framework discussed in section 3.

5 Optimising Hyper-parameters of Recurrent Neural Architectures

5.1 Experiments with Sequential MNIST

In this experiment, we trained DilatedRNNs with unpermuted and permuted sequential MNIST datasets in a classification task (classify digits 0-9 from their

[1] Enwik8 exhibits a single power-law due to the presence of XML code (strict markup grammar and long contextual dependencies). Consequently, the dependency decay curve of Enwik8 are different from the rest of the character-based datasets.

images). The original paper that introduced DilatedRNNs [3] used the same max dilation hyper-parameter for both of these datasets i.e. 256, and a standard set of dilations (i.e., $1, 2, 4, 8, \dots$). The best results reported by [3] for these two datasets were: unpermuted sequential MNIST 99.0/99.2 and permuted sequential MNIST 95.5/96.1. However, our analysis of these datasets has revealed different max dependencies across these dataset. For unpermuted sequential MNIST we identified a periodicity of 28 and so we expected the max dilation value to be near 28 to deliver better performance. In permuted sequential MNIST we identified that the dependencies extend up to 780 and so we would expect better performance by extending the max dilation up to this value.

Table 1. Results for sequential MNIST using GRU cells

# of Layers	Set of Dilations	Hidden per Layer	Accuracy
4	$1, 2, 4, 8$	20/50	98.96/99.18
5	$1, 2, 4, 8, 16$	20/50	98.94/99.21
6	$1, 2, 4, 8, 16, 32$	20/50	**99.17/99.27**
7	$1, 2, 4, 8, 16, 32, 64$	20/50	99.05/99.25
8	$1, 2, 4, 8, 16, 32, 64, 128$	20/50	99.15/99.23
9	$1, 2, 4, 8, 16, 32, 64, 128, 256$	20/50	98.96/99.17

Table 2. Results for permuted sequential MNIST using RNN cells

# of Layers	Set of Dilations	Hidden per Layer	Accuracy
7	$1, 2, 4, 8, 16, 32, 64$	20/50	95.04/95.94
8	$1, 2, 4, 8, 16, 32, 64, 128$	20/50	95.45/95.88
9	$1, 2, 4, 8, 16, 32, 64, 128, 256$	20/50	95.5/96.16
10	$1, 2, 4, 8, 16, 32, 64, 128, 256, 512$	20/50	95.62/96.4
11	$1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 780$	20/50	**95.66/96.47**

To test these hypotheses we trained DilatedRNNs with various sets of dilations. To keep our results comparable with those reported in [3] the original code[2] was kept unchanged except for the max dilation hyper-parameter. The test results of these experiments are in tables 1 and 2. For unpermuted task, the model delivered best performance for max dilation of 32. Focusing on the results of the permuted sequential MNIST, the best performance was delivered with the max dilation of 780. These results confirm that the best performance is obtained when the max dilation is similar to the span of the LDDs of a given dataset.

[2] `https://github.com/code-terminator/DilatedRNN`

5.2 Experiments with Character-Based Datasets

In this experiment, we trained DilatedRNN with PTB dataset in a language modeling task. The standard evaluation metric of language models is perplexity and it is a measure of the confusion of the model when making predictions. The original DilatedRNNs paper [3] reported best performance on PTB with a perplexity of 1.27 using GRU cells and max dilation of 64. However, the original DilatedRNN implementations used for this task has not been released. Hence we used another implementation of DilatedRNN[3]. Having done multiple experiments with this new implementation, including using the same hyper-parameters as those reported in [3], we found that the perplexity was always higher than the original implementation. We attribute this to the fact that [3] does not report all the hyper-parameter settings they used and so we had to assume some of the hyper-parameters (other than the dilations) used for this task. The test perplexity results of these experiments are in table 3.

Table 3. Results for character-based datasets (PTB) using GRU cells

Model #	# of Layers	Set of Dilations	Hidden per Layer	# of Parameters	Test Perplexity (bpc)
1.	7	$1, 2, 4, 8, 16, 32, 64$	256	2660949	1.446
2.	8	$1, 2, 4, 8, 16, 32, 64, 128$	256	3055701	1.471
3.	12	$1, 2, 4, 8, 16, 32, 64, 128, 256, \ldots$ $512, 1024, 2048$	256	4634709	1.561
4.	7	$1, 2, 3, 4, 5, 6, 8$	256	2660949	1.416
5.	8	$1, 2, 4, 8, 16, 32, 73, 240$	256	3055701	1.456
6.	12	$1, 2, 3, 4, 5, 6, 8, 11, 18, 32, 73, 240$	256	4634709	**1.386**

Recall from section 4.2, that the character-based PTB dataset exhibits a broken power-law dependency decay. Consequently, following the framework we presented, we expect that a model that has set of dilations that follow the pattern that we have proposed for broken power-laws (with different densities of connections on either side of the inflection point) should outperform a model that uses a standard set of dilations. This experiment was designed to test this hypothesis while controlling the model size. We designed six different model architectures (listed in table 3), three using the standard dilations (model# $1, 2,$ and 3) and three using dilation patterns informed by the dependency decay curve analysis of PTB dataset (model# $4, 5,$ and 6). Furthermore, in designing these architectures we controlled for the effect of model size on performance by ensuring that for each of the 3 models whose dilations were fitted to the dependency decay curve there was an equivalent sized model in the set of models using the standard pattern of dilations. Following the analysis of the PTB dataset we observe the inflection point of the broken power-law decay to be $\approx$12. We designed one model (model# 4) with a dense set of dilations up to the inflection point and two using

[3] `https://github.com/zalandoresearch/pytorch-dilated-rnn`

patterns of dilations that follow the broken power-law pattern. Of these two, one model (model# 5) used standard set of dilations up to the inflection point and sparse dilations beyond and the second model (model# 6) used the dilation patterns that we believe most closely fits the broken power-law dependence decay of PTB dataset. The results of these six models are listed in table 2.

One finding from these experiments is that larger models do not always outperform smaller models, for example models 1 and 4 outperform models $2, 3$, and 5. More relevantly, however, if we compare models of the same size but with different patterns of dilations i.e. comparing model 1 with 4, model 2 with 5, and model 3 with 6, we find that in each case the model that uses pattern of dilations suggested by our framework (models $4, 5$, and 6) outperforms the corresponding standard dilations model (models $1, 2$, and 3). Furthermore, the best overall performance is achieved by model 6 which uses a broken-power law pattern of dilations. The custom set of dilations used by model 6 maximizes it's memory capacity over the steeper region of the dependency decay curve before the inflection point by introducing dense dilations. However, beyond the inflection point the dilations are sparse allowing for longer dependencies to be modeled by using less memory capacity. We interpret these results as confirming that best model performance is obtained when the pattern of dilations is customized to fit the dependency decay curve of the dataset being modeled.

6 Discussion & Conclusion

In this paper. we have set out an approach to the design the hyper-parameter search space for recurrent neural architectures. The basis of this work is to use mutual information to analyse the dependency decay of a dataset. The dependency decay curve of a dataset is indicative of the presence of a specific dependency decay pattern in the dataset. For example, our analysis of sequential MNIST and character-based datasets indicate that the dependency decay of sequential MNIST is very different from character-based datasets. Understanding the properties of the underlying grammar that produces a sequence can aid in designing the grid-search space for the hyper-parameters to model a given dataset. In this work, we have used DilatedRNNs to illustrate our approach, and have demonstrated how customising the dilations to fit the dependency decay curve of a dataset improves the performance of the DilatedRNNs. However, we believe that the core idea of this approach can be more generally applied. Different neural architectures use different mechanisms to model LDDs, however these mechanisms have hyper-parameters that control how they function and we argue that it is useful to explicitly analyse the dependency decay curve of a dataset when selecting these hyper-parameters for the network.

Acknowledgments

The research is supported by TU Dublin Scholarship Award. This research is also partly supported by the ADAPT Research Centre, funded under the SFI

Research Centres Programme (Grant 13/ RC/2106) and is co-funded under the European Regional Development Funds. We also gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU under NVIDIA GPU Grant used for this research.

References

1. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks **5**(2), 157–166 (1994)
2. Campos, V., Jou, B., Giró-i Nieto, X., Torres, J., Chang, S.F.: Skip rnn: Learning to skip state updates in recurrent neural networks. In: International Conference on Learning Representations (2018)
3. Chang, S., Zhang, Y., Han, W., Yu, M., Guo, X., Tan, W., Cui, X., Witbrock, M., Hasegawa-Johnson, M.A., Huang, T.S.: Dilated recurrent neural networks. In: Advances in Neural Information Processing Systems 30, pp. 77–87 (2017)
4. Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., Salakhutdinov, R.: Transformer-XL: Attentive language models beyond a fixed-length context. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (2019)
5. El Hihi, S., Bengio, Y.: Hierarchical recurrent neural networks for long-term dependencies. In: Proceedings of the 8th International Conference on Neural Information Processing Systems. pp. 493–499. MIT Press, Cambridge, MA, USA (1995)
6. Graves, A., Wayne, G., Danihelka, I.: Neural Turing Machines. ArXiv e-prints (Oct 2014)
7. Hochreiter, S.: Untersuchungen zu dynamischen neuronalen Netzen. Master's thesis, TU Munich (1991)
8. Hochreiter, S., Bengio, Y., Frasconi, P.: Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In: Kolen, J., Kremer, S. (eds.) Field Guide to Dynamical Recurrent Networks. IEEE Press (2001)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
10. Kelleher, J.D.: Deep Learning. MIT Press, Cambridge, MA, USA (2019)
11. Lin, H.W., Tegmark, M.: Critical behavior in physics and probabilistic formal languages. Entropy **19**(7) (2017)
12. Mahalunkar, A., Kelleher, J.D.: Understanding Recurrent Neural Architectures by Analyzing and Synthesizing Long Distance Dependencies in Benchmark Sequential Datasets. arXiv e-prints arXiv:1810.02966 (Oct 2018)
13. Mahalunkar, A., Kelleher, J.D.: Using regular languages to explore the representational capacity of recurrent neural architectures. In: Artificial Neural Networks and Machine Learning – ICANN 2018. pp. 189–198. Springer (2018)
14. Merity, S., Xiong, C., Bradbury, J., Socher, R.: Pointer sentinel mixture models. CoRR **abs/1609.07843** (2016), `http://arxiv.org/abs/1609.07843`
15. Rhoads, J.E.: The dynamics and light curves of beamed gamma-ray burst afterglows. The Astrophysical Journal **525**(2), 737–749 (1999)
16. Salton, G.D., Ross, R.J., Kelleher, J.D.: Attentive language models. In: Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 441–450 (2017)
17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 5998–6008 (2017)