1 INTRODUCTION

In this master thesis the prospect of a recurrent neural network, RNN, with a stochastic structure varying in time to regularize, Monte Carlo RNN, MCRNN, is presented. Inspiration is drawn from biology where neural networks organically evolve their structure, formalized in the concept of a Markov process. How this algorithm is structured, however, is inspired from physics processes where energy differences drive the algorithm.

RNNs are a class derived from feedforward neural networks with iteration over the fixed structure to achieve temporal dynamics [18]. Forming along the time axis is a directed graph, achieving dynamic behavior in the sense of memory. Utilizing the memory to capture patterns also requires dynamical programming for the necessity of mapping long sequences to short. The RNNs excel at pattern recognition in data sequences e.g. natural language processing, machine translation and most commonly speech recognition [12]. However, launching RNNs has been slowed as despite the straightforward concept introduced in 1986 [17], implementing has been fraught with difficulties of capturing long-range temporal dependencies and of slow computation. With advances in hardware this problems have been somewhat mitigated.

Within deep learning the other competing model is the convolutional neural network, CNN. This class is most commonly applied to image recognition or analyzing visual imagery in general. However, as of late, a variation has been designed to compete with RNNs on processing sequences, the so called temporal CNN. Recent studies have been made showing superiority of said architecture over RNNs[9].

Initially RNNs were called dynamic recurrent neural networks, DRNs, displaying time-varying responses as per dynamical systems. During the early stages of development, a key issue was, shown by Bengio [1], that the necessary condition for the system to robustly store discrete state information for the long-term results in a sufficient condition of gradient decay. The issue of vanishing gradient [15] or exploding thereof when training RNNs, affecting long-range temporal dependencies limits the ideal application.

The phenomenon of vanishing gradient can be exemplified when utilizing the conventional training algorithm backpropagation through time, BPTT, first unfolding the network then backpropagating the error through the network [18]. Small changes permeate the network possibly causing exponential diminishing or increase of certain later weights in the network, affecting the gradient [25]. Additionally, with this naïve algorithm, unfolding and effectively producing a feed forward network, training would be impossible for a large number of iterations. An attempt at solving this is by truncating and not unfolding the whole network at every step, possibly using parallelization to calculate in parts. However, this would consequentially truncate long-term dependencies simultaneously. Seemingly a Mexican stalemate, efforts to remedy this have been somewhat successful.

One early method of mitigating the effects came with so-called NARX networks (Nonlinear AutoRegressive with eXogenous inputs method) [18] which introduce direct connections between current and past hidden states, reducing nonlinearities inherent in the RNN structure. But these architectures only alleviate by a constant factor the duration of temporal dependencies that can be learned [28].

A stochastic approach termed the echo state network, ESN, employs random connections but suffers from large structural overhead and is unable to cope with data-intensive problems. This stems from the more general approach of random weight guessing, RG, which suffers in complex structures only being reliable under solution-dense conditions. This could be used in benchmark evaluation for identifying faulty design. Moreover, the random structures impressive performance in certain domains suggests that it is worth investigating ESN-based initialization [28].

Leading RNN architecture is the long short-term memory [24], LSTM, which is currently in use in state-of-the-art speech recognition and smart assistants [21]. The LSTM use memory units, linear and adept at controlling information to and from the unit, productively truncating the gradient without undermining performance. There has been further development and improvement on the LSTM structure and several variants have been devised, tests have been conducted and an overall comparable performance was confirmed[3]. The Gated recurrent units, GRUs, akin to the LSTM but with fewer parameters is one commonly applied alteration but has been shown to be lesser than the LSTM in application. However, complex models require more training. As has been established LSTM has proceeded to solve pathological long-range temporal dependencies and was successfully applied to speech and handwritten text recognition [14, 13], robotic control [6]. Attempts have been made to progress further, putting LSTMs in a grid[5] or in combination with generative models and evaluation seems promising[4, 2, 11]. In conclusion, RNN and especially LSTM belong to the principally most general and powerful sequence processors out there.

LSTM has clear advantages over conventional RNNs, however, with the coming of Hessian free optimization, HF, in addition to structural dampening this edge is lessened. With the method RNNs proceed to solve pathological long-term dependencies, but computationally it still lacks in comparison [28].

From the RNN formulation above a problem with RNNs is the mapping of long to short sequences and this has been recently treated with attention-based models [7]. Earlier the solution consisted in compressing the input meanwhile losing information and spending resources compressing. With attention-based models came the implementation of variable length memory which gives flexibility in generating the output as well as accessing different parts of the memory at different output times.

Challenges ahead, besides the ones mentioned, include faster learning, representational capacity and weights as states [28]. Faster learning relates to the high cost of calculating the gradient on long sequences, methods to approximate derivatives under such circumstances quickly and efficiently are crucial. The two latter pertain to the size of the hidden state, the information and computation related to a sequence, by increasing the size it is possible to achieve longer-term temporal dependencies. This is where the stochastic nature of the proposed algorithm, changing the structure through time, aims to increases the size of the hidden state.

The thesis will consider a proof of concept of the introduced algorithm and regularization, using HF optimization for the RNN. Additionally, the algorithm will treat the standard data set MNIST. A possible novel application is the optimization of other neural networks when analyzed by the suggested structure.

1.1 MOTIVATION

Whenever a neural network, NN, is produced and benchmarked there is a so called "Brasklapp" that albeit the NN or NN related algorithm performing well there is a possibility that the hyperparameters regarding structure or other was not optimal. However, this is not well studied area. The implications of optimizing structure are a possible performance boost or faster training if a smaller NN could produce similar results. The task of hyperparameter optimization is daunting and fraught with nonlinearities and has been handled in one way by generating a set of architectures, such as incrementally increasing the hidden layer, and in parallel letting them go through the training procedure. This brute force method of testing all reasonable structures after training completely avoids the problem and relies heavily on hardware, it is not a scalable method as the size of the NN increases more computational power is required to in parallel train the NNs. The effort here is to streamline the procedure of generating architectures into MC-walk.

1.2 RELATED WORK

The idea of modifying the structure of the neural network in time is not unfamiliar and a related major breakthrough is the dropout technique [27]. Both algorithms can be seen as adding noise to hidden units, the proposed stochastic structure, MCRNN, takes this one step further by allowing mutation. Formally a regularization technique, dropout is an attempt to increase generalization performance by reducing overfitting during learning. Similar to MCRNN dropout instead tackles the challenge of overfitting by dropping units or nodes from the network, making the network simpler and thus easier to train. Additionally, this opens for the possibility of combining exponentially many different networks architectures efficiently, drawing from ensemble learning this constitutes a pseudo-ensemble. However, dropout cannot be applied in a straightforward manner to RNNs [10] but has to be limited to a certain subset[31] to be fruitful. The paper by Zaremba, Sutskever, and Vinyals [31] only treats the LSTM structure and claim that it could be successfully applied to any RNN.

Several other papers treat the issue by applying dropout to the connections in varying ways, limiting to the feedforward network, only the non-recurrent connections or the hidden states update vector. Further modification include the so-called Fracternal and Curriculum dropout or the stochastic depth, the last one consists of dropping an entire layer[16]. Combining the method of dropout and DropConnect [30], a generalization of dropout for large fully-connected layers within structures, and other methods resulted in a successful variant of the LSTM, namely the avereged stochastic gradient descent(ASGD) Weight-Dropped LSTM, AWD-LSTM for short. A landmark neural network architecture in language-modeling.

Another approach is the so called Zoneout[20], where instead of dropout a zoneout would revert the units activations to previous timestep. Injecting noise into the network, using a mask designating which units are to be affected. This has the benefit of keeping propagation of gradients efficient and preserving state information, akin to feedforward stochastic depth networks[1]. By optimally designating occurence of zoneout, state of the art advancements have been made[23]. In the article by Rocki, Kornuta, and Maharaj [23] a concept surprise is introduced, the discrepancy between the last state's prediction and target. Surprisal could further be an optimum way of determining the stochastic evolution of MCRNN. The authors Rocki, Kornuta, and Maharaj [23] conjecture that stochasticity leads to robustness in the hidden state. As mentioned, MCRNN is designed with a the intention of increasing the hidden state and implementation of surprise will be left out.

Weakening the stochastic activation of dropout in the sense of periodically managing information flow through the network is another method. One such model is the Phased LSTM [22] which specializes in asynchronous sensory events that carry timing information. Another is the Clockwork RNN [19], a NN architecture which partitions the hidden layer into distinct modules of temporal granularity, performing computations at its imposed clock rate. The proposed model takes one step into generalizing by delving into stochastic forming of the network itself.

Mentioned earlier was the attention-based model, first introduced by Bahdanau, Cho, and Bengio [8] improving on the encoder decoder-based approach which utilizes RNNs typically for machine translation. As RNNs and LSTMs still suffer from being unable to capture long data sequences, attention-based models are an effort to counter the problem by screening information. Weighing information based on a context given by the encoder, generating a sentence from a picture or translating a sentence with the decoder, usually an LSTM. The concept has evolved into altogether attention-based architectures without RNNs such as the Transformer[29] which has outperformed its predecessors. This introduced the notion of self-attention.

Of general importance is the universal approximation theorem, which states that a feedforward neural network is able to approximate any Borel measurable function on a compact domain. A proof has been done even for the case of RNN [26] and of interest is the particular evolving structure and its implications of the proposed MCRNN. However, this will not be studied in this thesis.

[1] Not to be confused with stochastic feedforward neural networks. Despite their alluding name, stochastic feedforward networks are more related to Restricted Boltzmann Machines, RBMs, which learn probability distributions over its set of inputs.

[1] Bengio et al. "Learning long-term dependencies with gradient descent is difficult." In: (1994).

[2] Chung et al. "A Recurrent Latent Variable Model for Sequential Data". In: (2015).

[3] Greff et al. "LSTM: A Search Space Odyssey". In: (2015).

[4] Gregor et al. "DRAW: A Recurrent Neural Network For Image Generation". In: (2015).

[5] Kalchbrenner et al. "Grid Long Short-Term Memory". In: (2015).

[6] Mayer et al. "A system for robotic heart surgery that learns to tie knots using recurrent neural networks". In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2006, pp. 543–548.

[7] Xu et al. "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention". In: (2015).

[8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2014. arXiv: 1409.0473 [cs.CL].

[9] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling". In: *CoRR* abs/1803.01271 (2018). arXiv: 1803.01271. URL: http://arxiv.org/abs/1803.01271.

[10] Justin Bayer et al. *On Fast Dropout and its Applicability to Recurrent Networks*. 2013. arXiv: 1311.0701 [stat.ML].

[11] Bayer and Osendorfer. "LEARNING STOCHASTIC RECURRENT NETWORKS". In: (2015).

[12] A. Graves et al. "A Novel Connectionist System for Improved Unconstrained Handwriting Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31 (2009), pp. 855–868.

[13] Alex Graves and Jürgen Schmidhuber. "Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks". In: *Advances in Neural Information Processing Systems 21*. Ed. by D. Koller et al. Curran Associates, Inc., 2009, pp. 545–552. URL: http://papers.nips.cc/paper/3449-offline-handwriting-recognition-with-multidimensional-recurrent-neural-networks.pdf.

[14] Graves and Schmidhuber. "Framewise phoneme classification with bidirectional LSTM and other neural network architectures". In: *Neural Networks* 18 (2005), pp. 602–610.

[15] Hochreiter. "Untersuchungen zu dynamischen neuronalen netzen". In: (1991).

[16] Gao Huang et al. "Deep Networks with Stochastic Depth". In: *CoRR* abs/1603.09382 (2016). arXiv: 1603.09382. URL: http://arxiv.org/abs/1603.09382.

[17] Williams R. J., Hinton Geoffrey E., and Rumelhart David E. "Learning representations by back-propagating errors". In: *Nature* 323 (1986), pp. 533–536.

[18] S. C. Kremer J.F. Kohlen. *A Field Guide to Dynamical Recurrent Networks*. Wiley-IEEE Press, 2001.

[19] Jan Koutník et al. *A Clockwork RNN*. 2014. arXiv: 1402.3511 [cs.NE].

[20] David Krueger et al. *Zoneout: Regularizing RNNs by Randomly Preserving Hidden Activations*. 2016. arXiv: 1606.01305 [cs.NE].

[21] Cade Metz. *Apple is bringing the AI revolution to your iphone*. Accessed: 2016-06-16. 2016.

[22] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. *Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences*. 2016. arXiv: 1610.09513 [cs.LG].

[23] Kamil Rocki, Tomasz Kornuta, and Tegan Maharaj. "Surprisal-Driven Zoneout". In: *CoRR* abs/1610.07675 (2016). arXiv: 1610.07675. URL: http://arxiv.org/abs/1610.07675.

[24] J. Schmidhuber S. Hochreiter. "Long Short-Term Memory". In: (1997).

[25] Hochreiter S. et al. "A Field Guide to Dynamical Recurrent Neural Networks. Gradient flow in recurrent nets: the difficulty of learning longterm dependencies". In: (2001).

[26] Anton Maximilian Schäfer and Hans Georg Zimmermann. "Recurrent Neural Networks Are Universal Approximators". In: *Artificial Neural Networks – ICANN 2006*. Ed. by Stefanos D. Kollias et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 632–640. ISBN: 978-3-540-38627-8.

[27] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

[28] I. Sutskever. "Training Recurrent Neural Networks". In: (2013).

[29] Ashish Vaswani et al. "Attention Is All You Need". In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[30] Li Wan et al. "Regularization of Neural Networks using DropConnect". In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, 2013, pp. 1058–1066. URL: http://proceedings.mlr.press/v28/wan13.html.

[31] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. "Recurrent Neural Network Regularization". In: *CoRR* abs/1409.2329 (2014). arXiv: 1409.2329. URL: http://arxiv.org/abs/1409.2329.