INDIAN INSTITUTE OF TECHNOLOGY

DELHI

MASTER THESIS

Recurrent Neural Networks as Cognitive Models

Author:

Rishubh Singh

Supervisor:

Sumeet Agarwal

A thesis submitted in fulfillment of the requirements

for the degree of Master of Technology

in the

Department of Computer Science

June 25, 2019

Declaration of Authorship

I, Rishubh Singh, declare that this thesis titled, "Recurrent Neural Networks as Cognitive Models" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

Acknowledgements

I would specifically like to thank Prof Sumeet Agarwal for supervising my research on this project and providing every help possible. I would like to thank the Machine Learning group at IIT Delhi for help throughout the project. I would also like to thank my family for the constant support and my brother for his guidance.

Abstract

The goal of our research is to create cognitive and biologically plausible recurrent neural networks for learning syntax-sensitive dependencies. Long short term memory (LSTM) are able to capture long range statistical regularities but are bad models of the brain in terms of architecture. We begin addressing this problem by using biologically plausible neural networks like the Excitatory-Inhibitory Recurrent Neural Network (EIRNN) (H. Francis Song, 2016) to model number agreement learning tasks and comparing it to LSTM. We probe the competence of multiple recurrent neural network architectures over a plethora of closely related tasks, which shows that RNNs are worse at remembering/finding locus of grammatical errors than capturing long range dependencies when compared to LSTMs. We further find relations between performance on similar tasks that shows expectation vs locality effects of providing more information. We also study ablation effects on LSTM which show that the input and forget gates are the most important for learning.

Contents

List of Figures

[1] H. Francis Song, 2016.

List of Tables

Chapter 1

Introduction

1.1 Context

Early works on Deep Learning have been made in the 1940-1960s, and they mainly describe biologically inspired learning : models like Hebbian learning (Hebb, 2001) and the Perceptron (Rosenblatt, 1958). The second wave called Connectionism came in 1960-1980s, with the invention of backpropagation which is still the backbone of for training most artificial neural networks today. The old connectionism had its roots emerging from various fields including philosophy, physiology (Thorndike's Connectionism, Hull's Learning Rule), neuropsychology (Hebbian Learning) and mathematics. In the seminal paper, *A logical calculus of the ideas immanent in nervous activity* McCulloch and Pitts explicitly laid out the foundations of neural modelling in terms of propositional logic with main aspects being static structure of the network and the activity of the neuron being an 'all-or-none' process.

Since the past 4-5 decades, however, neural networks have been approached from an engineering perspective, the designers only interested in making them as efficient as possible. Consequently, moving away from the biological inspiration, which is shown in the blunt assessment of connectionist research back in 1988 : *" The ultimate goals of AI and neuroscience are quite similar but that they have become obscured by erroneous epistemological assumptions drawn on the one hand from the arguments of Alan Turing and Alonzo Church about the universal*

problem-solving capabilities of computers (suggesting that the brain may be understood as a computer) ... These new approaches, the misleading label "neural network computing" notwithstanding, draw their inspiration from statistical physics and engineering, not from biology. "

Although we have been taking inspiration from the human brain every now and then, like the creation of convolutional neural networks from the idea of receptive fields (and connections in the visual receptive system), most research in the field of ML and AI recently has rather taken the engineering approach of making the performance of these systems better rather than the aspect of circling back to understand how it translates to modelling the human brain. Recent developments, in the past decade or two, have taken the approach of using these artificial neural networks (ANNs) with constraints as models of specific parts of the brain and shown that the activations of the units in the ANN have a close resemblance to actual neural activity recorded from humans / other test subjects. In contrast, ANNs have also been used to create models that resemble human performance on particular tasks. These two approaches of making biologically plausible ANN models of the human brain and creating models that resemble it behaviorally (performance wise) are the two major ways of studying neural mechanisms underlying cognitive functions.

Deep Learning has revolutionized the field of natural language understanding (NLP / NLU) as well as creating better frameworks for getting insights into neural circuitry. Recurrent neural models like the Vanilla RNN, GRU and LSTM have been the core of models that have produced state of the art results on almost all NLP tasks. Recent research has shown that RNNs with explicit biological constraints are great for modelling neural data from biological circuits (H. Francis Song, 2016), and that LSTMs have the ability to capture complex dependencies in natural language and their performance varies with difficulty as expected from humans (T Linzen, 2016).

We aim to create biologically plausible networks that are at par in performance with humans and other state of the art neural models, in an effort to find and explain the differences among different recurrent models, and to bridge the gap between ANNs and what we know about the human brain.

1.2 Contribution

This Master's Thesis introduces a number of contributions to different aspects of modelling cognition for learning syntactic-sensitive dependencies using recurrent neural networks and bridging the gap between biologically plausible neural networks and high level tasks like Natural Language Processing.

- In the first chapter we go over previous research in the field, and show how current neural network models, although have high accuracy, architecturally are not biologically plausible, i.e. they aren't close to modelling the brain.

- In the second chapter, we go over Excitatory-Inhibitory Recurrent Neural Network introduced by H. Francis Song, 2016, which is a better model of the brain with explicit constraints on connections and weights and analyze its performance on learning syntax sensitive dependencies compared to SRN (Simple Recurrent Network / Standard RNN) and LSTM (Long Short Term Memory).

- In the third chapter, we introduce an intermediate model AbLSTM and a set of tasks with incremental difficulty to understand linguistic effects and differences among different recurrent network models.

- In the last chapter, we provide qualitative and quantitative analysis between recurrent models and among different sets of tasks in an effort to provide a better understanding of cognitive models.

1.3 Recurrent Neural Networks

Recurrent networks have connections between nodes form a directed graph along a temporal sequence. Recurrent neural networks can be imagined by assuming the network to be a single unit which has an inherent state which keeps getting updated given the new input and the previous network state. Unrolling this network along the temporal direction gives us the directed graph visualisation that we generally come across. The storage state can further be replaced by another network or graph that incorporates time delays/feedback loops. Traditionally most commonly used recurrent neural architectures are Standard RNNs, LSTMs and GRUs.

1.3.1 Standard Recurrent Neural Network (RNN)

$$h_t = tanh(w_{ih}x_t + b_{ih} + w_{hh}h_{(t-1)} + b_{hh})$$

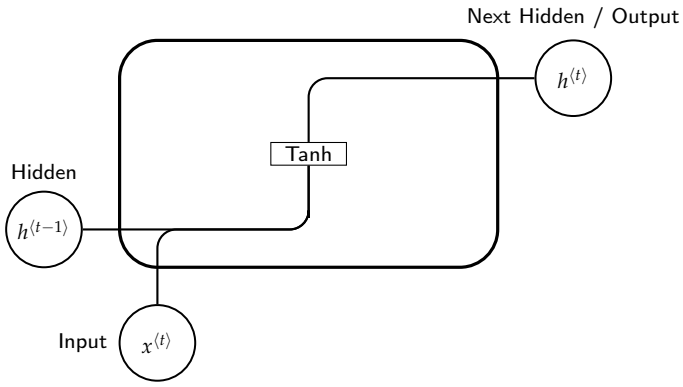


FIGURE 1.1: Standard Recurrent Neural Network (RNN)

1.3.2 Gated Recurrent Units (GRU)

$$z_t = sigm(w_{iz}x_t + b_{iz} + w_{hz}h_{(t-1)} + b_{hz})$$

$$r_t = sigm(w_{ir}x_t + b_{ir} + w_{hr}h_{(t-1)} + b_{hr})$$

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot tanh(w_{ih}x_t + w_{hh}(r_t \cdot h_{(t-1)}) + b_{hh})$$

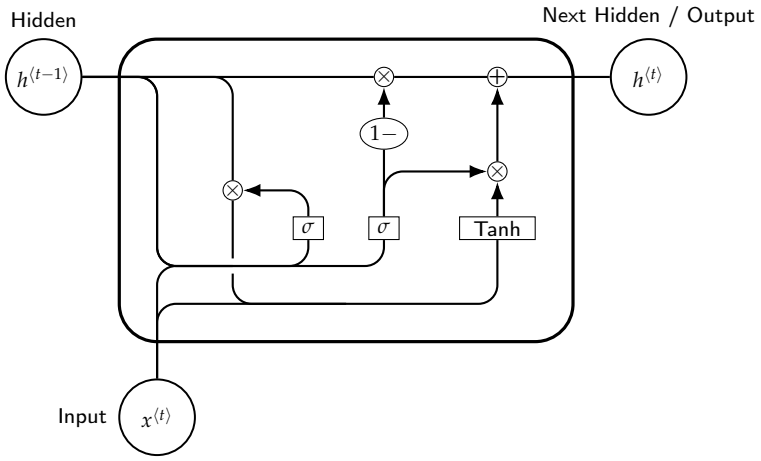


FIGURE 1.2: Gated Recurrent Unit (GRU)

1.3.3 Long Short Term Memory (LSTM)

$$i_t = sigm(w_{ii}x_t + b_{ii} + w_{hi}h_{(t-1)} + b_{hi})$$

$$f_t = sigm(w_{if}x_t + b_{i_f} + w_{hf}h_{(t-1)} + b_{hf})$$

$$g_t = tanh(w_{ig}x_t + b_{ig} + w_{hg}h_{(t-1)} + b_{hg})$$

$$o_t = sigm(w_{io}x_t + b_{io} + w_{ho}h_{(t-1)} + b_{ho})$$

$$c_t = f_t \cdot c_{(t-1)} + i_t \cdot g_t$$

$$h_t = o_t \cdot tanh(c_t)$$

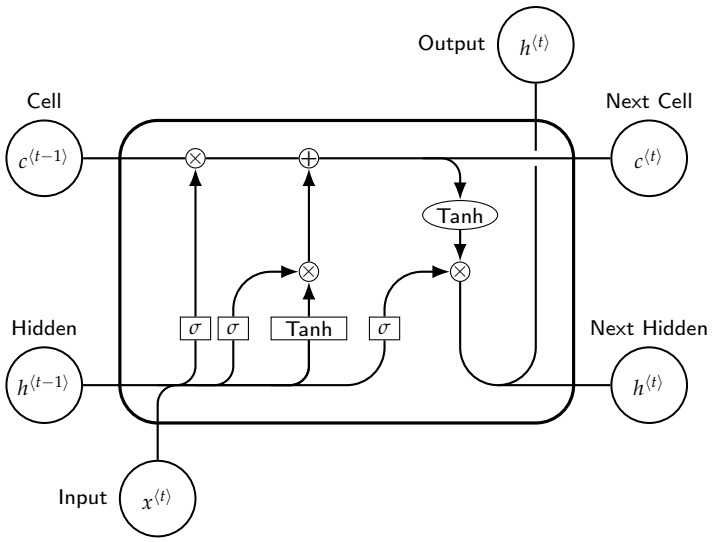


FIGURE 1.3: Long Short Term Memory (LSTM)

We also get the same performance with a combined *i-f* gate LSTM, where $i = 1 - f$ as an LSTM or with a GRU, so we won't be discussing them in detail separately.

1.3.4 RNNs vs LSTMs

The general notion is that LSTMs (and GRUs) are usually better in performance compared to RNNs due to the incapability of RNNs in capturing long-term dependencies while training them using gradient descent due to the vanishing gradient problem. The widely accepted fact is that apart from the above problem, most recurrent neural network models are similar in performance which has been observed across innumerable tasks.

LSTMs off late have been at the core of most state of the art models for most NLP tasks. Linzen, Dupoux and Goldberg in their paper *'Assessing the ability of LSTMs to learn syntax-sensitive dependencies'* showed that the performance of LSTMs with increasing difficulty varies according to what would be expected from a human as well on most aspects (T Linzen, 2016). They further show that error rates increase only slightly when we switch from number prediction/inflection to a more indirect supervision consisting only of sentence-level grammaticality annotations without an indication of the crucial verb. Although, the language model trained without explicit grammatical supervision performed worse than chance on the harder agreement prediction cases even with state-of-the-art language models. These results suggest that explicit supervision is necessary for learning the agreement dependency using this architecture, limiting its plausibility as a model of child language acquisition. They compare SRN (Simple Recurrent Network / Standard RNN) with LSTM on the number prediction task showing that SRN's make twice as many mistakes as LSTMs and have difficulty learning and

correctly predicting verb number with increasing number of agreement attractors.

However, LSTMs are not biologically plausible models of the brain, given the type of connections (Hadamard products) in the network and the architecture of the network itself (gates). RNNs on the other hand are much better models structurally and they can be thought of as a fully connected graph of neurons which is updated at each time step given the input and the previous state of the network. Linzen et al do not compare SRN's with LSTMs on grammaticality where we see a stark difference in performance with SRN's not being able to learn at all. We then try to bridge the gap artificial neural networks and brain models by bringing biologically inspired and plausible neural models (like the EIRNN described in the following section) and explicitly constrained RNNs in the group of recurrent models to learn syntactic dependencies.

1.3.5 Excitatory-Inhibitory Recurrent Neural Network (EIRNN)

H. Francis Song, 2016

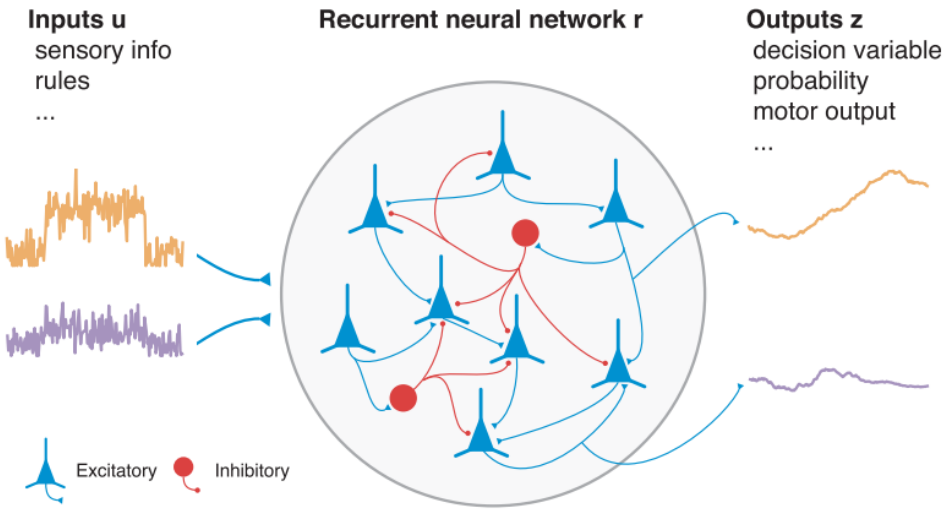


FIGURE 1.4: Excitatory-Inhibitory RNN[1]

Dale's Principle The basic and ubiquitous observation that neurons in the mammalian cortex have purely excitatory or inhibitory effects on other neurons. The analogous constraint that all connection weights from a given unit must have the same sign can have a profound effect on the types of dynamics, such as non-normality, that operate in the circuit. Moreover, connections from excitatory and inhibitory neurons exhibit different levels of sparseness and specificity, with non-random features in the distribution of connection patterns among neurons both within local circuits and among cortical areas. Notably, long-range projections between areas are primarily excitatory. Such details must be included in a satisfactory model of local and large-scale cortical computation.

EIRNN is similar to the standard RNN, except the fact that the input is same across all recurrences, which means it can be unfolded into a deep neural network with shared weights across all layers and an output at each layer, with explicit constraints on the network architecture and weights learned like the Dale's principle and/or other known information about structural constraints like imposing local only connectivity (instead of all neurons being interconnected). We also force the input and output connecting weight matrices to be non-negative to impose the constraint of long-range projections being excitatory only.

$$h_t = [\alpha \cdot (Dale(W_{rec})h_{(t-1)} + ReLU(W_{in})x)] + [(1-\alpha) \cdot h_{(t-1)}]$$

$$o_t = ReLU(w_{out}) \cdot h_t$$

where (assuming x is a 5×5 matrix)

$$Dale(x) = ReLU(x) \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

The ratio of excitatory to inhibitory connections is said to be close to 80-20% but can be treated as a hyperparameter for further analysis as needed.

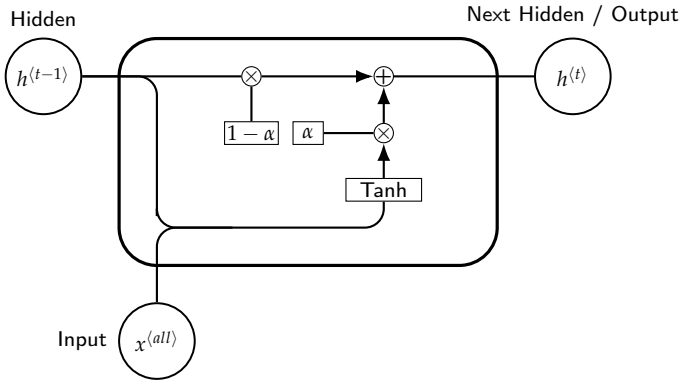


FIGURE 1.5: Excitatory-Inhibitory Recurrent Neural Network (EIRNN)

1.4 Syntax-Sensitive Dependencies

The form of an English third-person present tense verb depends on whether the head of the syntactic subject (in what follows, we'll refer to it as "the subject"), is plural or singular:

(1)
- The **pan is** on the stove.
- * The **pans is** on the stove.
- * The **pan are** on the stove.
- The **pans are** on the stove.

The subject and the corresponding verb are marked in bold and asterisks mark unacceptable sentences. While, in these examples, the subject and

the corresponding verb are adjacent, its not always the case. In such cases, agreement attractors[2] are underlined and intervening nouns with the same number as the subject is italicized.

(2) – The **pan** from the cupboard **is** on the stove.

Given a syntactic parse of the sentence, identifying the head of the subject corresponding to the verb and using that information to determine the number of the verb is simple. Although models that are insensitive to structure can run into difficulties capturing this dependency, one of the major issues being that there's no limit of the complexity of the subject NP, and any number of words can appear between the noun and the verb.

(3) – The **decision** of female Gothic writers to supplement true supernatural horrors with explained cause and effect **transforms** romantic plots and Gothic tales into common life and writing.

This property of the dependency makes it impossible to be captured by an n-gram model with a fixed n. LSTMs, a particular class of RNNs are able to capture these kinds of dependencies well across different kinds of tasks presented by T Linzen, 2016, that are explained below. However, from a cognitive science perspective, the LSTM architecture isn't biologically plausible, therefore the question of capability of RNNs to capture this dependency still remains.

The potential presence of agreement attractor entails that the model must identify the head of the syntactic subject that corresponds to a given verb in order to choose the correct inflected form of that verb.

1.4.1 The Number Prediction Task

The basic task to understand the extent to which sequence models can learn to be sensitive to the hierarchical structure of natural structure is *number*

[2]Intervening nouns with the opposite number from the subject head.

prediction. In this task, the model sees the sentence up to but not including a present-tense verb, e.g. :

(4) – The pan from the cupboard _____

The model then needs to predict the number of the following verb (SINGULAR or PLURAL). In order to perform well on this task, the model needs to encode the concepts of syntactic number and syntactic subjecthood: it needs to learn that some words are singular and others are plural, and to be able to identify the correct subject and correctly identifying the subject that corresponds to a particular verb often requires sensitivity to hierarchical syntax. Table 1.1 shows the distribution of sample types from the test set of 1.5 million samples in the corpus created from Wikipedia by T Linzen, 2016 [3].

	With Attractors	**With Intervening Nouns**
$n = 0$	92.7%	80.8%
$n = 1$	5.7%	12.7%
$n = 2$	1.1%	4.2%
$n = 3$	0.3%	5.1%
$n = 4$	0.1%	2.1%

TABLE 1.1: Corpus statistics of the T Linzen, 2016 number agreement test dataset (1.5M sentences)

1.4.2 The Inflection Task

The *inflection* task is similar to *number prediction*, the only difference being that the network receives the singular from of the upcoming verb also, apart from the words leading up to the word. The network then needs to decide between the singular and plural forms of the particular verb (*plays* and *play*). Having access to the semantics of the verb can help the network identify

[3] With Attractors : At least one attractor between the noun-verb pair. With Intervening Nouns : At least one noun between the noun-verb pair.

the noun that serves as its subject without using the syntactic subjecthood criteria. For example, in the following sentence :

(5) – People from the capital often eat pizza.

only people is a plausible subject for eat; the network can use this information to infer that the correct form of the verb is eat is rather than eats. This objective is similar to the task that humans face during language production: after the speaker has decided to use a particular verb (e.g., write), he or she needs to decide whether its form will be write or writes (Levelt et al., 1999; Staub, 2009).

1.4.3 The Grammaticality Task

The previous objectives explicitly indicate the location in the sentence in which a verb can appear, giving the network a cue to syntactic clause boundaries. They also explicitly direct the network's attention to the number of the verb. As a form of weaker supervision, we experimented with a *grammaticality* judgment objective. In this scenario, the network is given a complete sentence, and is asked to judge whether or not it is grammatical.

To train the network, we made half of the examples in our training corpus ungrammatical by flipping the number of the verb [4]. The network reads the entire sentence and receives a supervision signal at the end (ungrammatical sentences are marked with an asterisk) :

(6) – The **key** to the cabinet **is** on the stove.

– * The **keys** to the cabinet **is** on the stove.

[4] In some sentences this will not in fact result in an ungrammatical sentence, e.g. with collective nouns such as group, which are compatible with both singular and plural verbs in some dialects of English (Huddleston and Pullum, 2002); those cases appear to be rare.

1.5 Baseline and Generic Models

The words are encoded as one-hot vectors which are embedded in a 50-dimensional vector space. The recurrent network reads those words in a sequence (EIRNN reads the entire input at once); the final state of the network is fed into a logistic regression classifier which in our case is a fully connected layer from the final state of the network to two predicted values each predicting the confidence of one of the binary labels. Then the final prediction is the label with the higher predicted confidence. The input sentence is pre-padded with zeros to a fixed length of 50 words. The model is then trained [5] in an end-to-end fashion, including the word embeddings [6].

1.6 LSTMs with Explicit Syntactic Information

Adhiguna Kuncoro, 2018 show that the ability of LSTMs in capturing long range dependencies is dependent on capacity of the model and can be learned with high enough capacity. Experimenting with the capacity of LSTMs, they show that the accuracy in capturing longer dependencies using language models increases with increasing number of hidden units as shown in Table 1.2. They further find that models with access to explicit syntactic information do not improve on accuracy, but when the model architecture in determined by syntax (Recurrent Neural Network Grammars : RNNGs), number agreement is improved. This however further moves away from the idea of biological plausibility but gives an idea that hierarchical parsing might be of importance.

[5] Cross-entropy loss is used and the network was optimized using Adam (Kingma and Ba, 2015). We trained the number prediction model 5 times with different random initializations, and report accuracy averaged across all runs. The models have been trained with a batch size of to maintain consistency of learning rate and other parameters for having level base for comparison among models.

[6] The size of the vocabulary was capped at 10000 (after lower-casing). Infrequent words were replaced with their part of speech (Penn Treebank tagset, which explicitly encodes number distinctions); this was the case for 9.6% of all tokens and 7.1% of the subjects.

	n=0	**n=1**	**n=2**	**n=3**	**n=4**
Random	50.0	50.0	50.0	50.0	50.0
Majority	32.0	32.0	32.0	32.0	32.0
LSTM, $H = 50^{\gamma}$	6.8	32.6	≈ 50	≈ 65	≈ 70
LSTM, $H = 50^{\kappa}$	2.4	8.0	15.7	26.1	34.65
LSTM, $H = 150^{\kappa}$	1.5	4.5	9.0	14.3	17.6
LSTM, $H = 250^{\kappa}$	1.4	3.3	5.9	**9.7**	13.9
LSTM, $H = 350^{\kappa}$	**1.3**	**3.0**	**5.7**	**9.7**	**13.8**

TABLE 1.2: Error Rate for Number Prediction using LSTM for Language Modelling : γ specifies the reported result from T Linzen, 2016 and κ specifies the reported results from Adhiguna Kuncoro, 2018

Chapter 2

Learning Syntax Sensitive Dependencies using EIRNN

2.1 Model and Setup

As a first step, we tried modelling the *number prediction* task using EIRNN. We were making available to the model, the entire sequence of words provided at input, instead of a word by word input, like in RNNs/LSTMs. The idea was that the recurrent part of the network while seeing the input evolves to produce the final output.
The EIRNN model we use can be described using the following recurrence relation :

$$h_t = [\alpha \cdot (Dale(W_{rec})h_{(t-1)} + ReLU(W_{in})x)] + [(1-\alpha) \cdot h_{(t-1)}]$$

$$o_t = ReLU(w_{out}) \cdot h_t$$

α is a hyperparameter with 0.005 giving the best performance. Each input to the model is made to be a fixed length sentence (size 50 in our case) by pre-padding with zeros. It is then converted a sequence of vectors using word2vec, where the embeddings are learned while training the model through backpropagation. Hence, the sizes of the matrices and input used in

the above recurrence are :

$$x : 50 \times embedding_{size}$$

$$W_{in} : hidden_{units} \times 50$$

$$W_{rec} : hidden_{units} \times hidden_{units}$$

$$W_{out} : output_{units} \times hidden_{units}$$

We keep the embedding size to be 50, the number of output units to be 10 and vary the number of hidden units to see how the performance of the model varies. Hence, at each temporal during the model processing one sentence, the network has a state defined by hidden unit number of neurons, each being a vector of embedding size.

We also train a sequential input version of EIRNN : RNN Dale, and an ablated model of LSTM : AbLSTM. We describe and discuss AbLSTM in later chapters. The RNN Dale model is just imposing the Dale's constraint on the recurrent matrix of a standard RNN :

$$h_t = tanh(w_{ih}x_t + b_{ih} + Dale(w_{hh})h_{(t-1)} + b_{hh})$$

2.2 Experiments

On the *number prediction* task, EIRNN performs surprisingly well given lack of sequential input like in standard recurrent models. We get the best accuracy of 94.1% on the number prediction task with 15 hidden units (a hidden capacity of 750). The accuracy of the model decays on both increasing/decreasing the number of hidden units. As another striking fact, we also find the model being able to learn to an accuracy of 92.8% with just 3 hidden units. Although the capacity of the recurrent network in this case is still 150,

the model only has 3 *neurons (units)* that update each others state which is still giving a relatively high accuracy. The results of our experiments and comparison with the LSTM model are tabulated in Table 2.1.

Model	Accuracy (%)
EIRNN (3)	92.8
EIRNN (15)	94.1
EIRNN (50)	93.5
RNN (50)	97.7
RNN Dale (50)	97.8
AbLSTM (50)	98.0
LSTM (50)	98.7

TABLE 2.1: Overall Accuracy on the Number Prediction task

The performance of RNN Dale and RNN are almost same with accuracy difference being $< 0.2\%$ across different runs. Both RNN and RNN Dale have high overall accuracy and close to that of LSTM. We also train a model we call *AbLSTM (Ablated LSTM)* where we remove the i and f gates from the LSTM, and it performs equally well on the number prediction task as well. We will describe and discuss AbLSTM in later chapters.

On the *inflection* task as well, the EIRNN performs similar to what it does on number prediction, and so do other models, with some showing slight increase in overall accuracy owing to the additional information present given the singular form of the verb as shown in Table 2.2, which is as expected given the added information and resembles human performance.

Model	Accuracy (%)
EIRNN (3)	92.5
EIRNN (15)	94.2
EIRNN (50)	93.3
RNN (50)	97.9
RNN Dale (50)	98.0
AbLSTM (50)	98.1
LSTM (50)	98.9

TABLE 2.2: Overall Accuracy on the Inflection task

2.3 EIRNN Results and Comparisons to LSTM

Although the performance of EIRNN is commendable given it's architecture isn't inherently sequential like a standard RNN or LSTM, further analysis reveals it is heavily reliant on non-syntactic cues unlike the LSTM. Since the RNN (Standard and Dale), has a competitive performance to the LSTM on similar parameters, the lack of sequential processing appears to be the main reason of the shortcomings of EIRNN.

While, the performance of LSTM goes down gradually by 5-6% at the distance between the verb and the corresponding noun increases from 1 to 14, for EIRNN it is 17-18%, which is substantially higher as seen in Fig. 2.1. As we see from Fig. 2.2 and Table 2.3, EIRNN is very susceptible to intervening nouns, specifically agreement attractors. As the number of attractors between the verb and the corresponding noun increases, the drop in accuracy for EIRNN is more than that for LSTM. In the case of the last intervening noun being an attractor, the accuracy of EIRNN drops by 40-42% compared to 5-6% for the LSTM, showing the fact that because of lack of explicit sequential information, EIRNN ignores syntactic cues and uses the closest preceding noun to the verb very often.

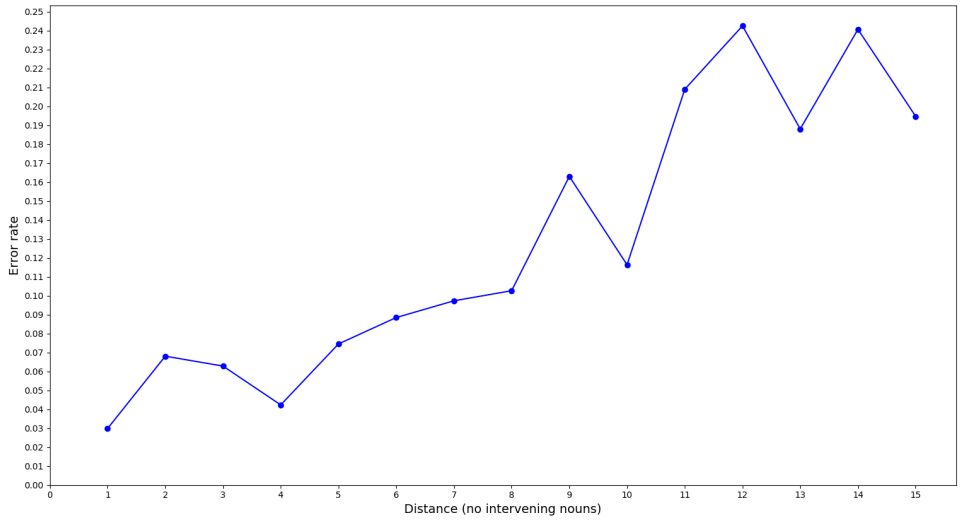


FIGURE 2.1: Error Rate vs Distance for EIRNN (no intervening nouns)

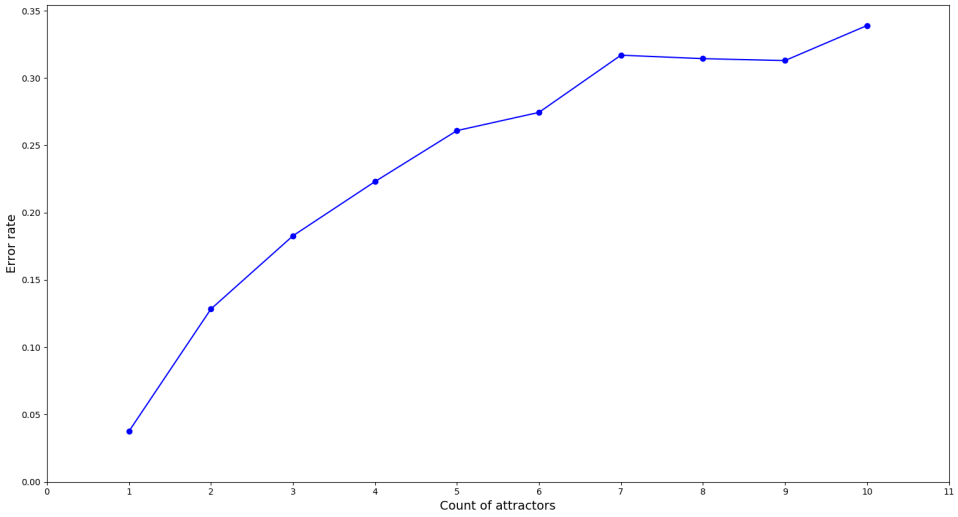


FIGURE 2.2: Error Rate vs Number of Attractors for EIRNN

Subject	No Last Intervening	Singular Last Noun	Plural Last Noun
Singular Subject	2.14 (0.31)	2.95 (0.48)	41.03 (3.93)
Plural Subject	7.15 (1.67)	44.43 (7.53)	5.74 (1.86)

TABLE 2.3: Error Rates (%age) for EIRNN. LSTM results in parentheses.

Although, EIRNN is able to separate out singular and plural nouns and verbs and learn well separable embeddings as seen from plotting the embeddings onto their first two principal components in Fig 2.3.

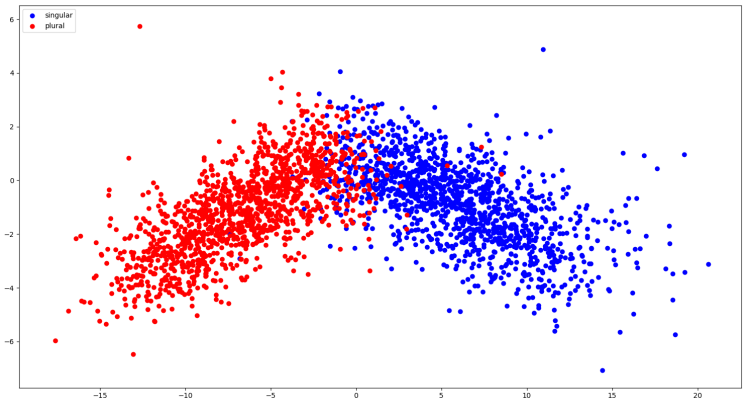


FIGURE 2.3: Embeddings of singular and plural nouns and verbs, projected onto their first two principal components.

Chapter 3

Grammaticality and Ablated LSTM

RNNs/RNN Dale's and LSTMs exhibit a huge difference in performance over the *grammaticality* task, where the LSTM is able to model grammaticality with an accuracy as high as 95-96%, while the RNN is not able to learn anything and perform equivalent to a random selector. To further explore and understand the differences between these models and the reasons for those differences, we create :

- AbLSTM (Ablated LSTM), a model architecturally between RNN and LSTM.
- A new task *'Grammaticality Plus'*, based on grammaticality with configurable difficulty.

The new model and task help us observe, analyze and understand performance barriers between different models.

3.1 Ablated LSTM : AbLSTM

$$g_t = tanh(w_{ig}x_t + b_{ig} + w_{hg}h_{(t-1)} + b_{hg})$$

$$o_t = sigm(w_{io}x_t + b_{io} + w_{ho}h_{(t-1)} + b_{ho})$$

$$c_t = f \cdot c_{(t-1)} + i \cdot g_t$$

$$h_t = o_t \cdot tanh(c_t)$$

where i and f are learnable vectors.

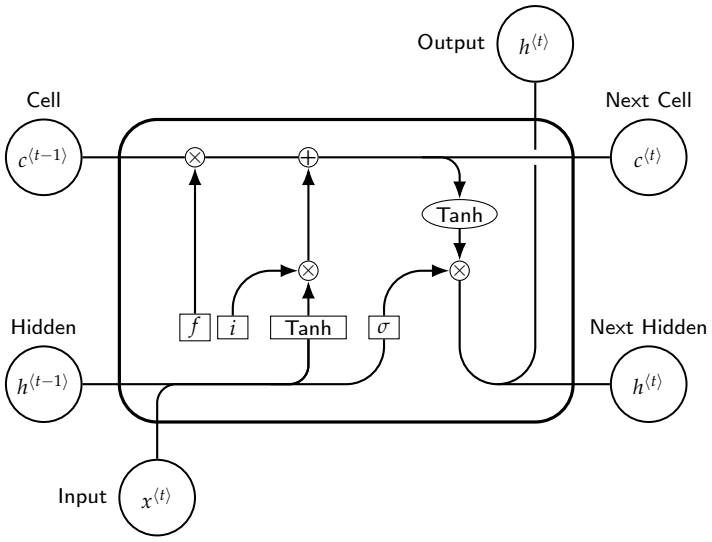


FIGURE 3.1: Ablated Long Short Term Memory (AbLSTM)

The difference between an ablated LSTM (AbLSTM) and LSTM is the replacement of i and f gates with learn-able vectors that are independent of the input and hidden state. Using scalars instead of vectors or other combinations of similar features performs similarly.

3.1.1 Modelling Grammaticality

Surprisingly, AbLSTM is unable to model grammaticality and the peak achieved performance stays close to random (49-53%) depending on initialization. The rather surprising observation that has been till now, is that all models that haven't been able to perform on grammaticality have failed doing so completely and only achieved random accuracy at best as summarized in Table 3.1. We'll also notice this trend going to the *grammaticality plus* task.

The performance of RNN and RNN Dale is comparable and almost the same on all tasks we test them on, hence either name interchangeably from now on to refer to both simultaneously.

Model	Accuracy (%)
EIRNN (3)	≈ 50
EIRNN (15)	≈ 50
EIRNN (50)	≈ 50
RNN Dale (50)	≈ 50
AbLSTM (50)	≈ 50
LSTM (50)	95.5

TABLE 3.1: Overall Accuracy on the Grammaticality task

3.2 Grammaticality on Half Sentences

On setting *alpha* in *grammaticality plus* task to 0, we get a task that is the similar to *number prediction* and *inflection*, in the sense that the input provided to the model is only until the verb. The set of objectives comprising of *number prediction*, *inflection* and *grammaticality zero*, explicitly indicate the location in the sentence in which a verb can appear (the following word after the input), giving the network a cue to syntactic clause boundaries. On the *grammaticality zero* task hence, we get decent performance by models other than the LSTM, with a high of 88/8% with EIRNN using 15 hidden units, 97.8% with RNN and 97.1% with the AbLSTM. All the results are summarized in Table 3.2.

Model	Accuracy (%)
EIRNN (3)	84.3
EIRNN (15)	88.8
EIRNN (50)	86.3
RNN Dale (50)	97.8
AbLSTM (50)	97.1
LSTM (50)	98.3

TABLE 3.2: Overall Accuracy on the Grammaticality task

Given that the models (except LSTM) are successfully able to model *grammaticality zero* (and *number prediction/inflection*) but unable to achieve above random accuracy on the full *grammaticality* task, there appears to be the singular reason behind the performance difference that they are not able to remember the locus of the error when they model the full *grammaticality* task.

3.3 Grammaticality Plus

As the next step, we test our hypothesis of the memory of the locus of the error being the problem. The *grammaticality plus* task is a clear extension to *grammaticality zero* and the intermediate between it and the full *grammaticality* task. We define a variable α that specifies the maximum number of words in the input sentence after the verb that is in context. We term the collection of these tasks as *'Grammaticality Plus'* with varying α creating a new task, with increasing difficulty as α in increased.

(7)
- $\alpha = 0$: The **version** of the chronicle that the annalists were working was written in different places at different times; the earliest evidence for one of its authors **places**
- $\alpha = 3$: The **version** of the chronicle that the annalists were working was written in different places at different times; the earliest evidence for one of its authors **places** it in Iona
- $\alpha = 6$: * The **version** of the chronicle that the annalists were working was written in different places at different times; the earliest evidence for one of its authors **place** it in Iona sometime after 563

3.4 Experiments

We ran experiments on LSTM, RNN Dale, AbLSTM and EIRNN (sentence input) varying values of α and capacity of model to observe how the test accuracy and validation accuracy over the entire training varied with both the variables. We also plot gradient norms for the parameters of the model as a marker for vanishing/exploding gradients. Later we also analyze, how the model's prediction changes with each input word and other parameter/intermediate values to understand what/how the model's learning.

3.4.1 Performance Comparisons

LSTM

As expected, LSTM is able to model the *grammaticality plus* task across all values of α. We test for α varying from 0 to 8 and the the full *grammaticality* task, which we have already looked at, and see the test accuracy of the model being above 98% for $\alpha = 0$ and decreases slightly as we increase α with a test accuracy of 95.5% for the full *grammaticality* task.

AbLSTM

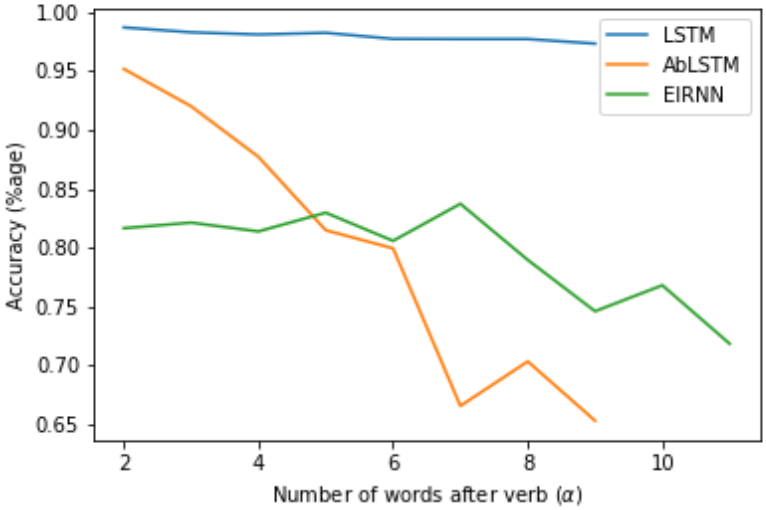
Training AbLSTM for α value higher than 1 is harder than LSTM or RNN and both the training and validation (and test) accuracies were falling suddenly after a certain number of iterations and the model becomes erratic which we have been unable to fix yet. Thus, there's a still a possibility of AbLSTM getting higher test/train accuracies than we see here if there wasn't the erratic drop in performance during training. We get a rather smooth decreasing trend in accuracy on varying α with AbLSTM, possibly due to the problem in training.

EIRNN

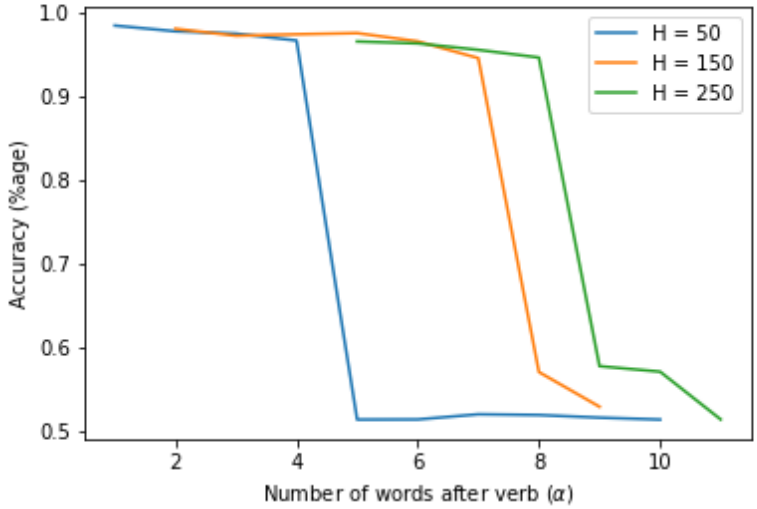
Unlike other models that get accuracies higher than 95% on *Half Grammaticality*, the accuracy of EIRNN is 88.8%, much lower than others. Although, the change in accuracy on increasing α has a general trend of decreasing gradually overall unlike the RNN which has a high accuracy until a specific value of α after which it drops to random. This pattern occurs because although the EIRNN has difficulty in modelling syntactic hierarchy well, it has access to the entire sentence at all times unlike the RNN.

RNN Dale

A rather unusual trend is observed with RNN on varying α. With 50 hidden units, the high test accuracy above 90% until $\alpha = 4$, and drops to random from $\alpha = 5$ onward. With 100 hidden units the test accuracy is high until $\alpha = 6$, and with 150 hidden units its high until $\alpha = 7$, and until $\alpha = 8$ for 250 hidden units. The sudden drop in accuracy from above 90% to random (50%) is rather unusual and striking.



(A) LSTM, AbLSTM and EIRNN



(B) RNN (50, 150 and 250 hidden units)

FIGURE 3.2: Test Accuracy on Varying α

3.4.2 Learning Pattern Analysis

LSTM

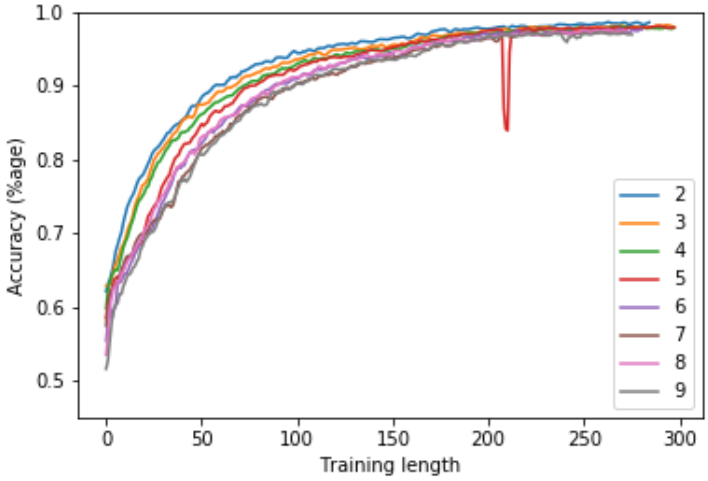
The validation accuracy curves are as expected given the test accuracy for each value of α. The higher the value of α, the curve shifts down slightly compared to the previous one as in Figure 3.3a. Each step of training in the figure corresponds to training over 3000 samples (the training set had approximately 158k samples).

AbLSTM

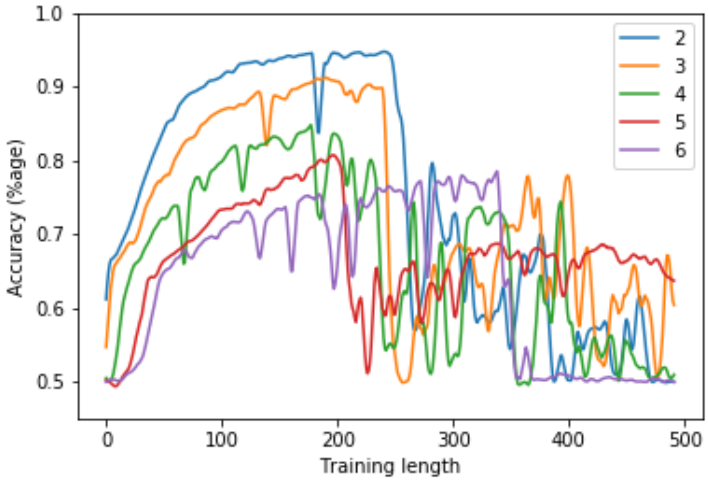
Training AbLSTM for α value higher than 1 is harder than LSTM or RNN and both the training and validation (and test) accuracies were falling suddenly after a certain number of iterations and the model becomes erratic which we have been unable to fix yet. This can be easily seen from the validation accuracy curves for AbLSTM (Fig 3.3b). Apart from the being erratic, the curves experience a sudden fall in accuracy within the second epoch of the training and are not able to learn again. The separation between curves for consecutive α's is also significantly more when compared to RNNs or LSTMs.

RNN Dale and EIRNN

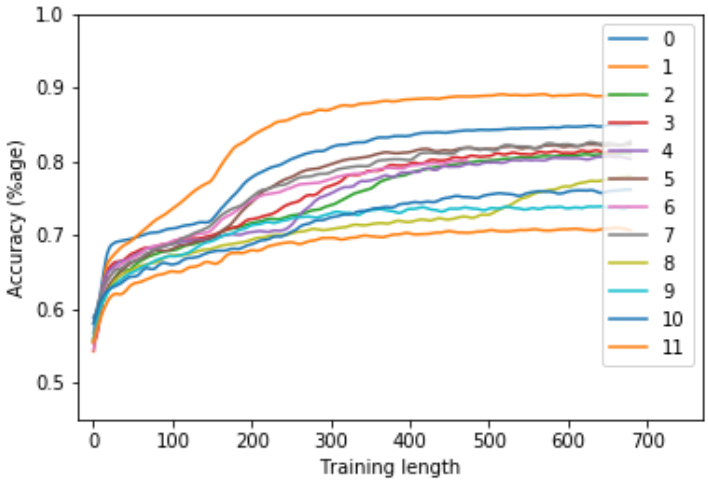
We observe the same downward shift of validation accuracy curve on increasing α for both RNN Dale and EIRNN. We also observe a rather strange simultaneous sudden dip and recovery in accuracy for higher capacity models with 150 and 250 hidden units which coincides with the abrupt drop in gradient norm as seen in Fig 3.5c. We need to look at gradient norm plots for other models and other datasets to understand and explain this phenomenon. In contrast to RNN Dale, the EIRNN model achieves an accuracy of approximately 70% at a high α of 11, significantly higher than RNN which has random accuracy at the same difficulty level (Fig 3.3c.



(A) LSTM

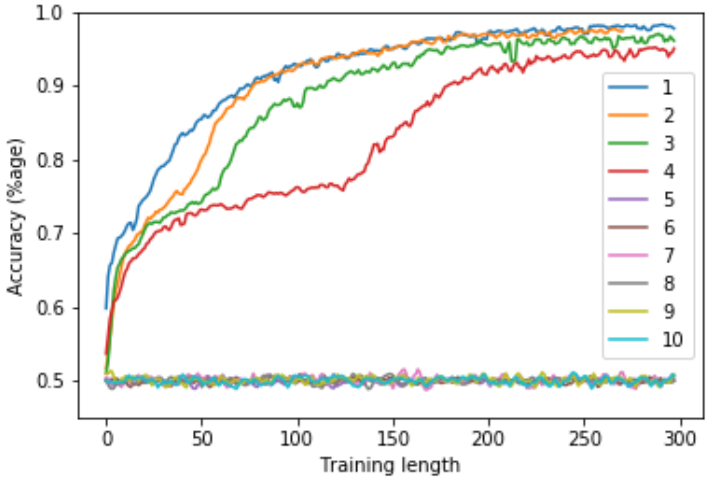


(B) AbLSTM

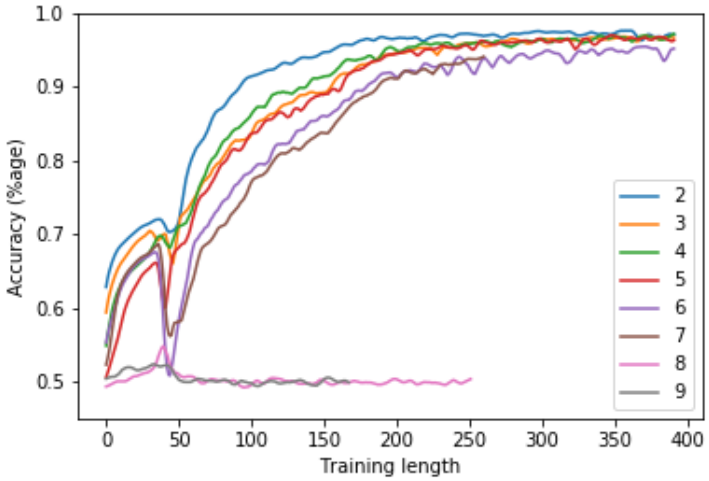


(C) EIRNN

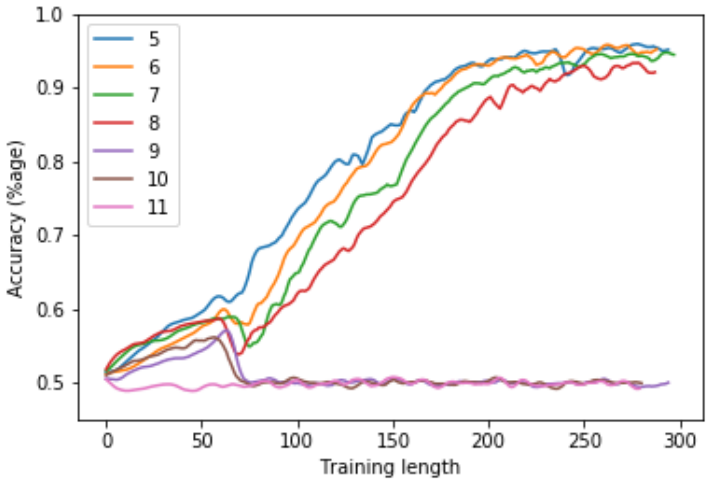
FIGURE 3.3: Validation accuracy vs training length for different α



(A) RNN : 50 hidden units



(B) RNN : 150 hidden units

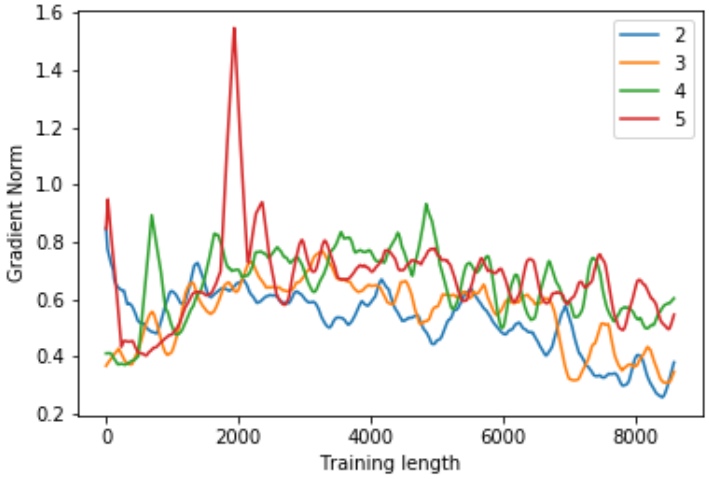


(C) RNN : 250 hidden units

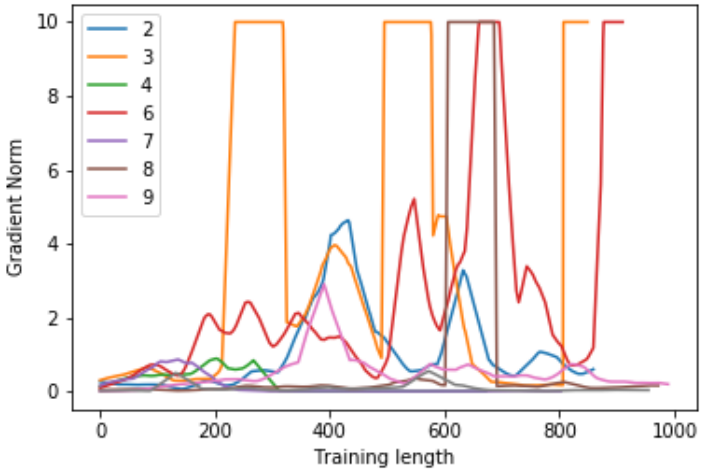
FIGURE 3.4: Validation accuracy vs training length for different α

3.4.3 Gradient Analysis

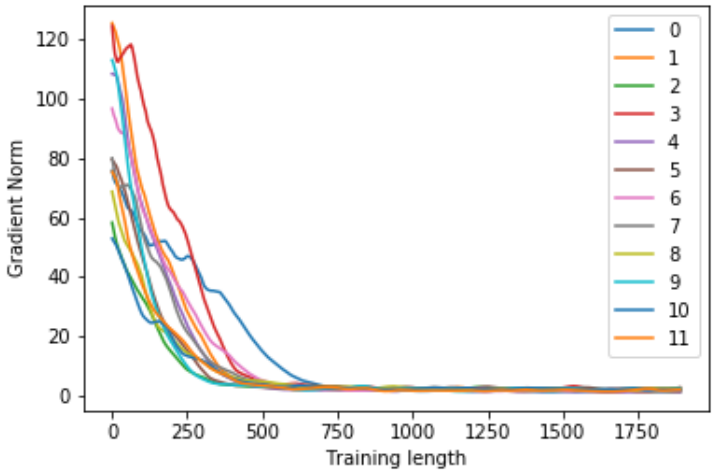
We plotted the gradient norms of all parameters of the neural network to study the vanishing gradient problem. The gradient norms for the recurrent matrix (W_{rec}), shows a clear difference among the four models : LSTM, RNN Dale, EIRNN and AbLSTM and also differences within RNN Dale at different capacities as well. While the gradient norms almost overlap for all values of α for LSTM showing consistent gradients, for RNNs (50 hidden units) the gradient norm start out high for all values of α, but for α greater than 4 then decline and stay negligible compared to other curves. For AbLSTM the inconsistent nature can be seen stemming from the gradient itself, the gradient norm curves being highly irregular. The first steep change for each curve overlaps with the sudden drop in accuracy. For EIRNN, the gradient norms start out very high, but decline to the same level of ≈ 2, which is the trend for the gradient norms of other parameters from the start (Appendix A).



(A) LSTM

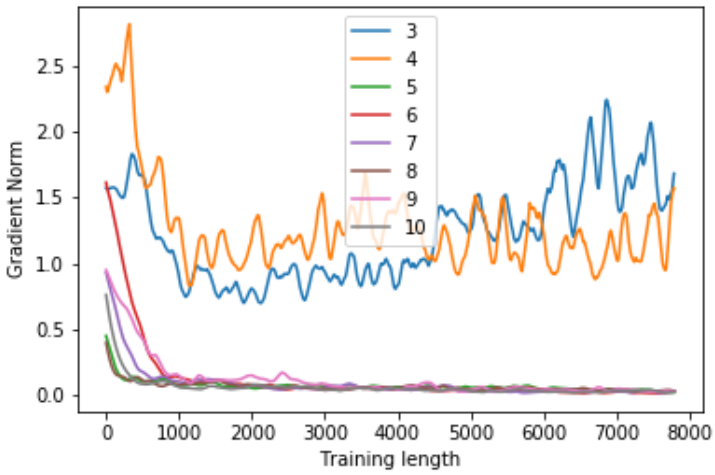


(B) AbLSTM

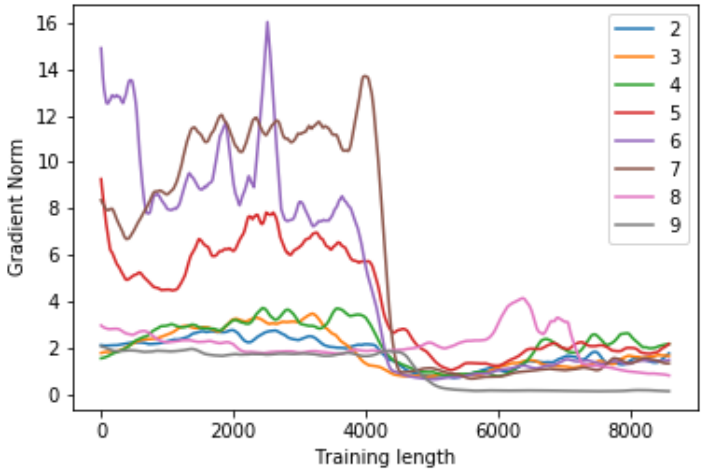


(C) EIRNN

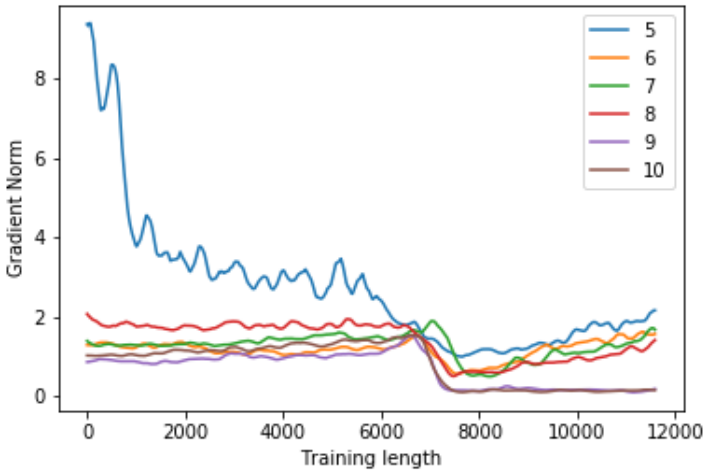
FIGURE 3.5: Gradient Norms for W_{rec} : recurrent matrix



(A) RNN (50)



(B) RNN (150)



(C) RNN (250)

FIGURE 3.6: Gradient Norms for W_{rec} for different capacity RNNs

3.5 Observations and Results

We analyze the activations of the hidden units and the prediction of different models as the network goes through the sentence. During testing, we pass the hidden state at each input word through the MLP (logistic regressor), to get the prediction and the confidence value (difference in prediction) for each model.

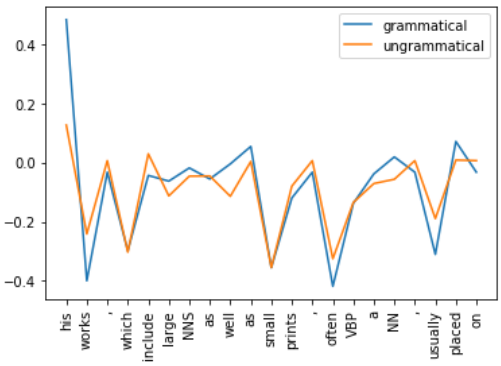
Testing the following inputs on the *Grammaticality Plus 6* task[1] :

(8) 1. His **works**, which include large NNS(murals) as well as small prints, often **VBP(depict)** a NN(teapot), usually placed on

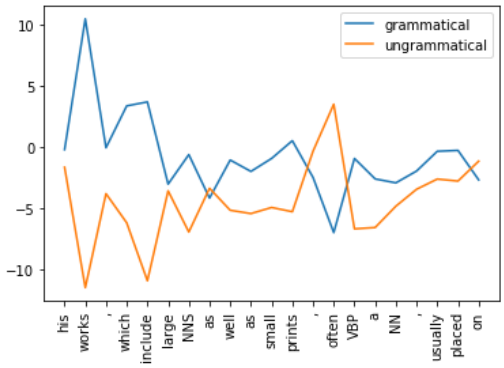
2. In a statement issued by the white house, the president said the American **people stand** united with the people of Russia

we observe that the predictions of EIRNN remain the same from the first iteration to the 20th. This pattern is consistent across the test samples except some exceptions where the two confidence curves (one for grammaticality and ungrammaticality each) cross each other ones (Appendix A). The confidence curves for LSTM are more steady than for other models except EIRNN. The models are expected to remain consistent with their prediction after seeing the verb in context, which is usual case with LSTMs. RNN with 250 hidden units, which is able to model this task (unlike RNN with 50 hidden units) also exhibits this pattern but there are a lot of cases in which it does not. AbLSTM also behave the same as RNN (250) mostly. More than 80% of the times, when the models get their prediction wrong, the difference in confidence values for grammatical and ungrammatical is low compared to when they get it right (apart from EIRNN). More plots on other sentences can be found in the Appendix A.

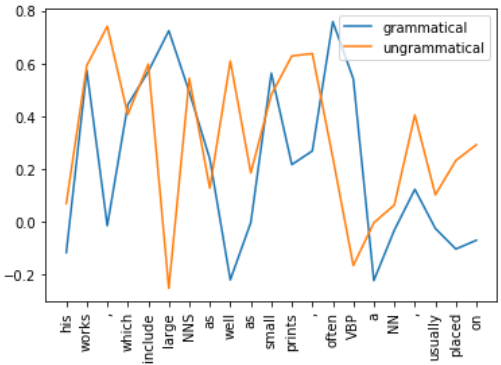
[1]Infrequent words in the corpus were replaced with their POS in the sentence for input to the model. Such words are mentioned in brackets beside their POS Tags in the sentence.



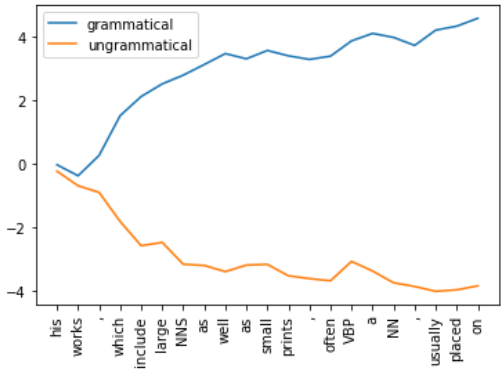
(A) RNN (50)



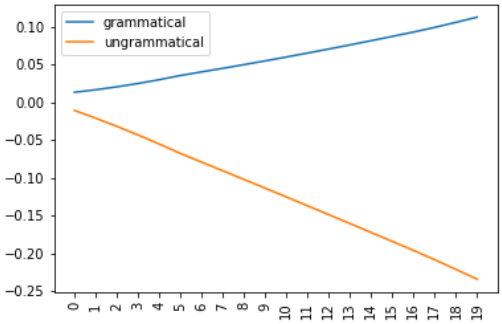
(B) RNN (250)



(C) AbLSTM

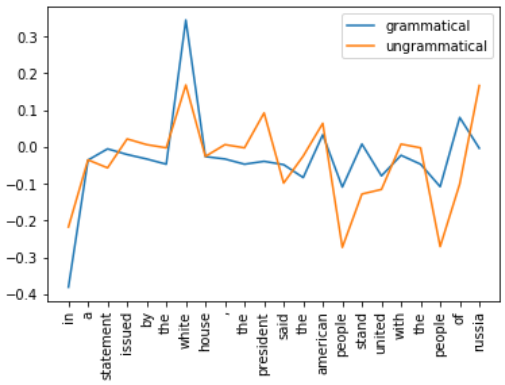


(D) LSTM

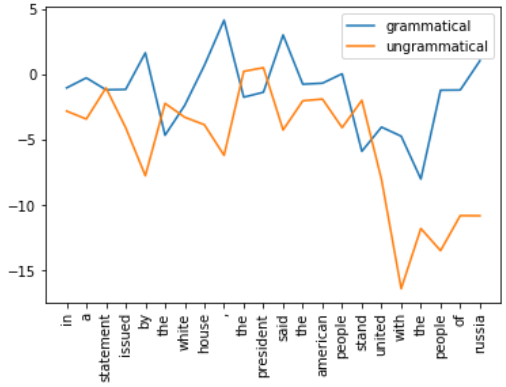


(E) EIRNN

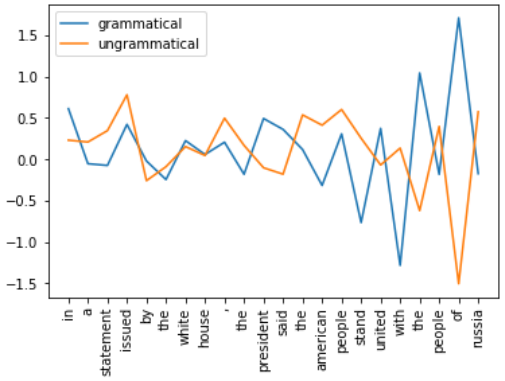
FIGURE 3.7: Plus 6 : Model prediction as the network goes through the input : Sentence (7)-1



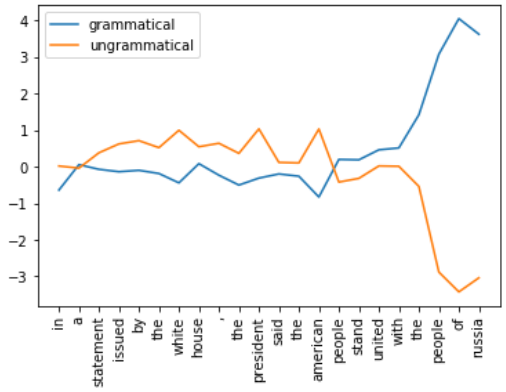
(A) RNN (50)



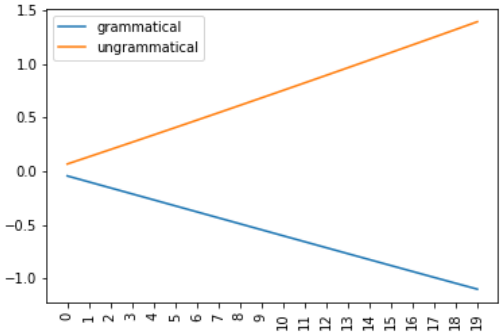
(B) RNN (250)



(C) AbLSTM



(D) LSTM

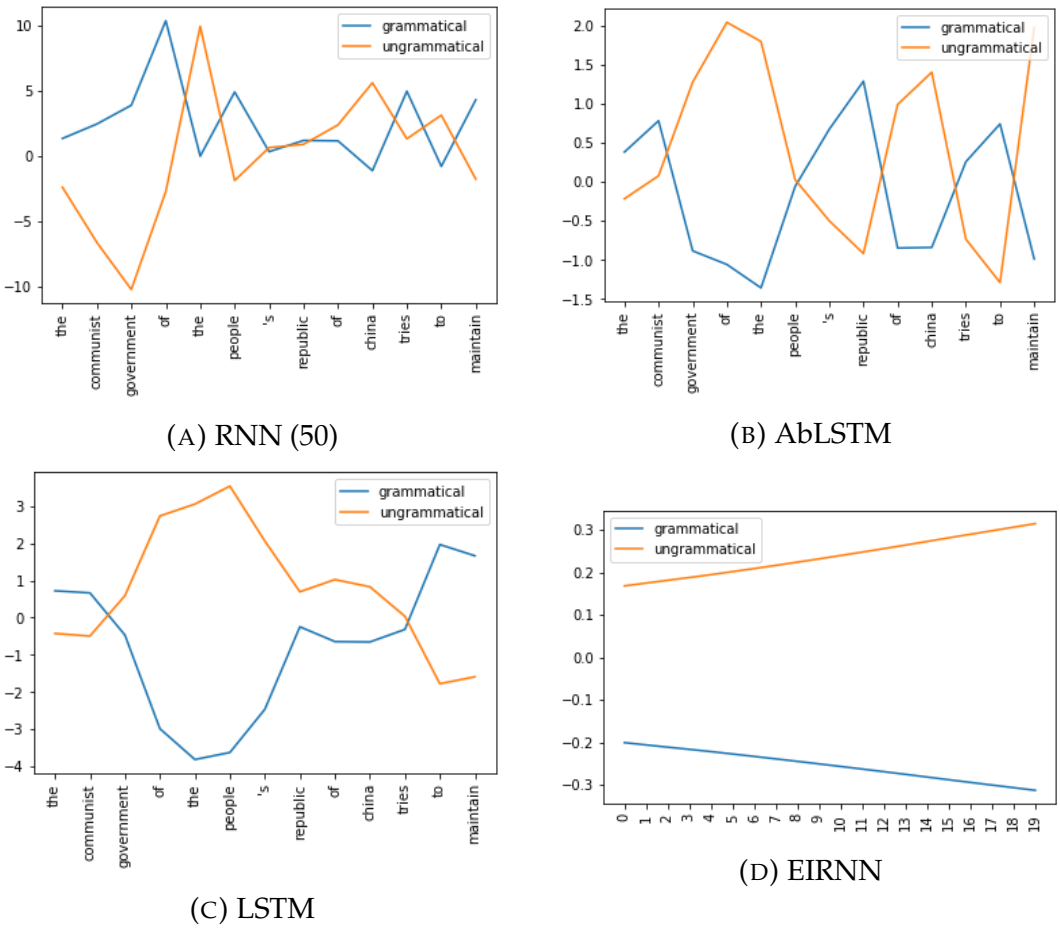


(E) EIRNN

FIGURE 3.8: Plus 6 : Model prediction as the network goes through the input : Sentence (7)-2

Doing the same analysis for *Grammaticality Plus 2* task, the results and observations are similar as that for *Plus 6*.

(8) 1. The communist **government** of the People's Republic of China **tries** to maintain



(A) RNN (50)

(B) AbLSTM

(C) LSTM

(D) EIRNN

FIGURE 3.9: Plus 2 : Model prediction as the network goes through the input : Sentence (8)-1

Chapter 4

Analysis and Conclusion

4.1 Qualitative Analysis

We manually examined over 500 samples to compare LSTM and RNN modelling results on multiple *Grammaticality Plus* tasks. Statistics for these results have been summarized in Table 4.1 [1].

	Plus 2	***Plus 3***	***Plus 6***
Both Correct	96.7%	96.4%	95.0%
Only LSTM Correct	1.7%	1.8%	2.9%
Only RNN Correct	0.8%	1.1%	1.2%
Both Wrong	0.7%	0.7%	0.9%

TABLE 4.1: Statistical comparison of LSTM and RNN on the *Grammaticality Plus 2, 3 and 6* tasks

Many of the errors by either models or both are attraction errors as noticed by T Linzen, 2016 as well. We mention here five classes of errors[2] of which three are similar to T Linzen, 2016[3], while also providing additional insights into the corpus, and the task itself.

[1] Tasks Plus 2 and Plus 3 data is on 50 hidden units' RNN and Plus 6 data is on 250 hidden units' RNN

[2] Examples presented from the task *Grammaticality Plus 3*.

[3] The analysis presented by **Linzen** is on the *Number Prediction* task, while our analysis is over *Grammaticality Plus* tasks.

The networks often misidentified the heads of noun-noun compounds. In (9)-1, for example, both the models predict the sentence to be grammatical even though the number of the subject football stadiums should be determined by its head stadiums [4]. This suggests that the networks didn't master the structure of English noun-noun compounds.

(9)
1. * Two relatively large **football *stadiums*** , Fawcett stadium in Canton and Paul Brown Tiger stadium in Massillon , **is** still in use
2. **Tax *collectors* collect** gold from guilds
3. * The **credits *sequence*** of Bob's Burgers **feature** the Belcher family
4. The urban services district encompasses the 1963 boundaries of the former city of Nashville , and the general **services *district* includes** the remainder of

Some errors appear to be due to difficulty not in identifying the subject but in determining whether it is plural or singular. In (10), in particular, there is very little information in the left context of the subject headquarters suggesting that the writer considers it to be singular:

(10) – Its current **headquarters is** in Rabat, Morocco

Some verbs that are ambiguous with plural nouns or nouns in general seem to have been misanalyzed as plural nouns and consequently act as attractors. The models predicted (11) as ungrammatical incorrectly, possibly because of the ambiguous verbs flies.

(11) – Every **arrow** that flies **feels** the pull of

As in this task, the models do not have an explicit cue to syntactic boundaries, they make mistakes misidentifying the noun-verb pair correctly. The

[4]The noun-noun compound subject is in bold and the head of the subject is italicized

task which itself becomes harder on addition of more noun-verb pairs, before the noun in context or in between the noun-verb pair on which the sample is testing grammaticality. Example (12) highlights some of these errors.

(12)
- I believe that the usual **way** a *tachometer works* **is** by spinning a
- * The **region** where the *stream originates* **are** in the highlands
- * The **state** that the *narrator wants* **are** seemingly a state

A much difficult task for the model is to understand inherent plurality introduced by group creation by using "commas" and conjunctive words likes "and" and "plus", in which case the head of the subject's number is not relevant is some cases. Both RNN and LSTM make mistakes in such sentences some of which are shown below in (13).

(13)
- * **Corn , soybeans, and wheat is** three common crops
- **Characters and plot are** complementary – they
- * **Health , arts and imaging technology, respiratory care and dental hygiene is** some of the

An unlikely but evident problem is also the ability of the model to distinguish proper nouns from other nouns when the sentence in lower case which removes one of the major clues present in general in text (14).

(14)
- The *United States* **system requires** that these differences
- * The credits **sequence** of *Bob's Burgers* **feature** the Belcher family
- * The *Blocks* **editor use** the *Open Blocks* software

4.2 Quantitative Analysis

We compared the test samples that each model (LSTM/RNN/AbLSTM) got wrong, which revealed that more than 50% of the samples which LSTM gets

wrong, AbLSTM also gets wrong across multiple *Plus α* tasks. On the other hand, RNN only gets around 20-30% samples wrong which LSTM predicted incorrectly. The above correlation shows that :

- Architectural closeness between AbLSTM and LSTM is reflected in behavioral (performance) closeness.
- The high correlation is errors made by different models shows that some sentences are inherently harder to model than others.

Apart from the types of sentences stated in the previous section that are harder for these recurrent networks to model, the known factors of distance and attractors affecting number agreement prediction is evident from average numbers in table 4.2 which shows that the models are more accurate on short range dependencies and that LSTMs are slightly better at long range dependencies than RNNs.

	Plus 2	***Plus 3***
Both Correct	2.35	2.35
Only LSTM Correct	5.72	5.3
Only RNN Correct	5.0	4.63
Both Wrong	6.81	6.58

TABLE 4.2: Average noun-verb distance comparison between RNN and LSTM on Plus 2 and Plus 3 tasks

To further analyze the difference in performance across *Grammaticality Plus* tasks, we compared the performance and specific errors of RNN on Plus 1 and Plus 3 and of LSTM on Plus 3 and Plus 6. For accurate analysis we only compare predictions on the same sample sentence with the same label (grammatical or ungrammatical) on both tasks. Contrary to expectations, the errors made by either RNN or LSTM on the assumed to be easier task, Plus 1 and Plus 3 for RNN and LSTM respectively, is not a subset of the errors

made on the harder task, Plus 3 and Plus 6 respectively. The surprising fact is that the two error sets (for both RNN and LSTM) have only around 40-45% overlap.

More than 50% of the samples that RNN got wrong on Plus 3, it correctly predicted on Plus 1 and the other way around as well. The same analysis on LSTM gives similar results. Although the overall performance decreases with increasing α, the errors made by the network are not consistent.

These results agree with the expectation vs locality effect trade-offs in linguistics and cognition. In modelling the harder task, the model has a more difficult task given the added words, but the added words in other cases help the network better parse the sentence, with multiple studies suggesting that strong expectations cancelling locality effects, which helps the model correctly predict a sample in Plus α which it predicted incorrectly in Plus $\alpha - 1$. This effect is even more prominent in samples where the word following the verb in context is another verb, while having more words following the verb would avoid confusion is predicting noun-verb pairs.

4.3 Discussion and Future Work

Neural network architectures have reached high accuracies in tasks based of natural language, LSTMs being the core of most of the state-of-the-art models. Although, a lot of previous research has debated the cognitive plausibility of these models and have tried to get closer in learning and performance to humans but architectural plausibility has not been explored yet. Comparison of different recurrent models of different linguistic tasks can provide insights into both cognition and understanding how architecture effects performance.

Since the majority of natural language sentence are grammatically simple, models can achieve high overall accuracy using flawed heuristics that fail

on harder cases (Socher, 2014), which is why EIRNN without sequential input does not perform well. But, an SRN with the same constraints as in EIRNN (RNN Dale) has a performance comparable to LSTMs in cases it is able to learn. The reason behind the inability of SRN or RNN Dale to suddenly be unable to model a task with slightly higher difficulty is unclear. Further experiments with different models and a different dataset are need to establish robustness and provide insights into this affect.

The ablated LSTM without the input and forget gates is unable to model grammaticality tasks well, which points to the fact that having the power of dynamically choosing what to remember and what to forget is provided by these gates without which harder tasks cannot be learned. Further analysis of the activation of these gates and different models that are more powerful than AbLSTM but less powerful than LSTM would shed further light on how architecture effects learning in LSTMs.

Changing the recurrent network to use the last two state instead of just one reduces the effective distance between two points in the sequence in half which could help in modelling. Taking an insight from the flawed success of EIRNNs, having access to the entire sentence helps in modelling dependencies. LSTMs with explicit window memories to access previous states/inputs could explain explicit vs implicit memory effects. This would also help in sentences in which the parse of the sentence cannot be created by reading words only sequentially.

A low correlation in errors made by a model on tasks that differ by the number of maximum words that follow the locus of the grammatical mistake in indicative of the fact that extra information can help in finding the noun-verb pairs in the sentence by inherently helping the model create a better representation of the sentence. This is similar to expectation and locality effects studied in psycho-linguistics to understand when and if more information hurts or helps. Further linguistic analysis on the corpus and the error sets of

the models across multiple tasks would provide more concrete evidence.

The bigger picture question is whether syntax parsing is necessary for learning number agreement and even other more complicated linguistic tasks or are sequences enough and syntax trees are for just our understanding. The question translates into modelling these tasks using neural networks when we talk of cognitive plausibility that by modelling language using LSTM/RNNs in a sequence, are we showing that explicit syntactic information is unnecessary or the neural network itself ends up creating hidden representations like the syntax trees while learning such tasks. Are we modelling the brain or are we modelling the way the brain processes information using these recurrent neural networks.

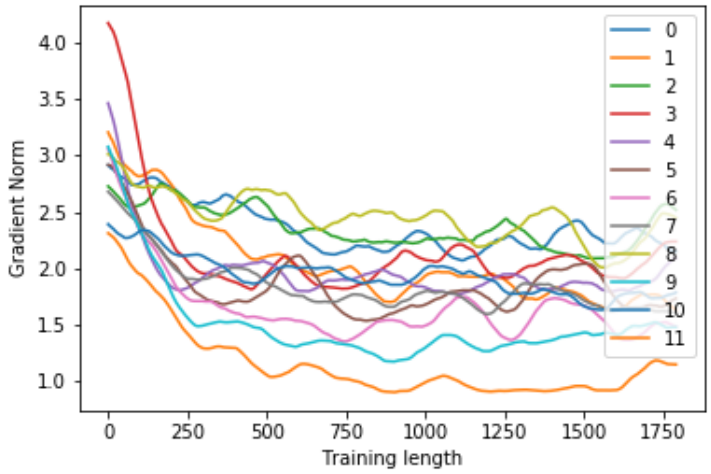
4.4 Conclusion

Summary of Contributions : In this Master's Thesis we studied recurrent neural networks as plausible cognitive models on dependency tasks and looked into the biological plausibility of the architecture of these models.

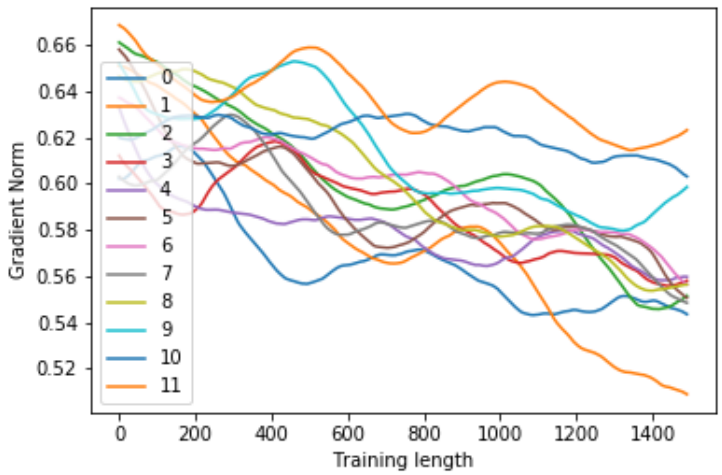
- We analyzed the performance of architecturally plausible neural networks with explicit biological constraints.
- We provided insights into differences in different recurrent neural networks and factors that affect them.
- We showed relationships between different tasks based syntax-sensitive dependencies and that more information can both help and hurt in a linguistic setting.
- We proposed insights that inspire models that are biologically plausible and can bridge the gap between in architecture when we talk of cognitive plausibility.

Appendix A

Figures

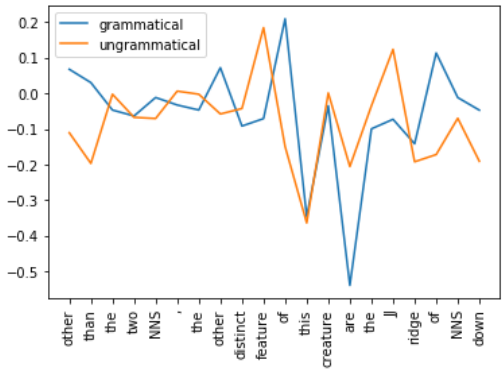


(A) W_{in}

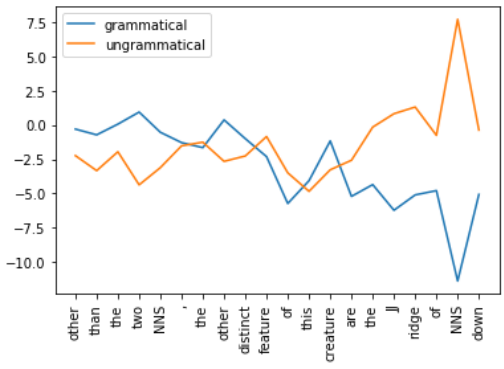


(B) W_{out}

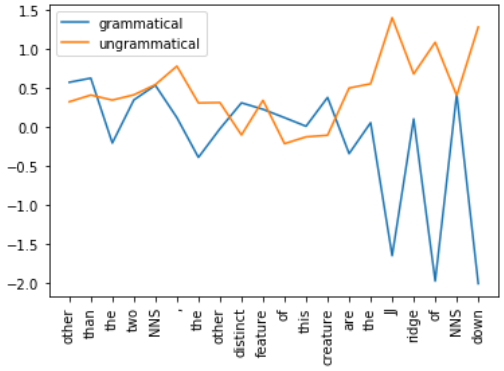
FIGURE A.1: Gradient Norm Plots for W_{in} and W_{out} for EIRNN



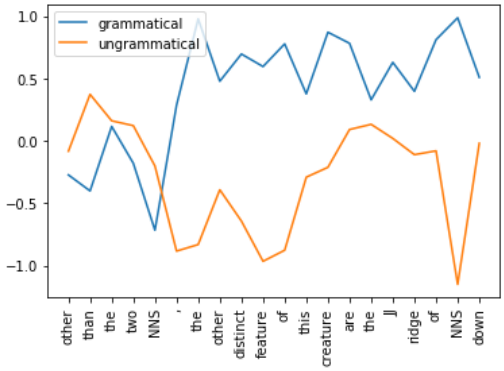
(A) RNN (50)



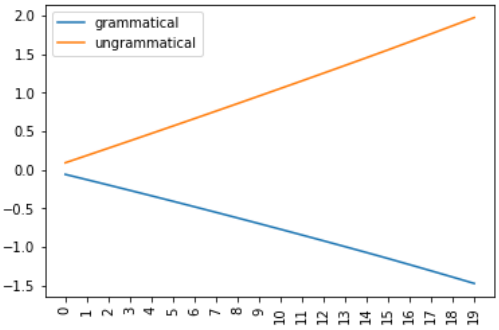
(B) RNN (250)



(C) AbLSTM

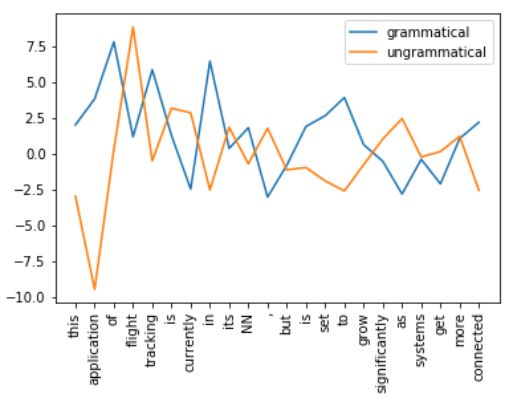


(D) LSTM

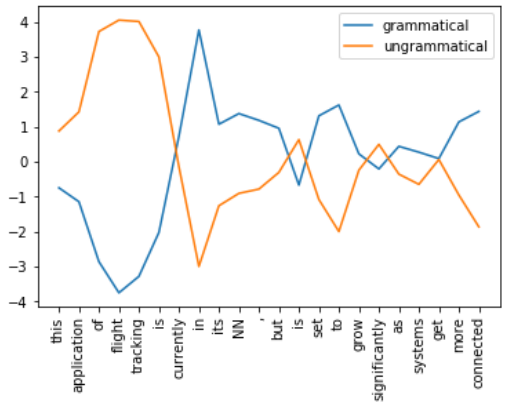


(E) EIRNN

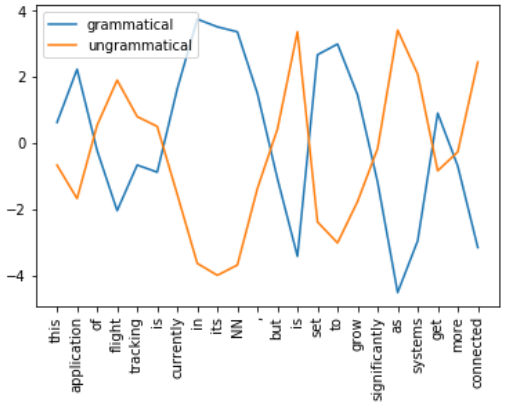
FIGURE A.2: Plus 6 : * Other than the two NNS, the other distinct **feature** of this creature **are** the JJ ridge of the NNS down



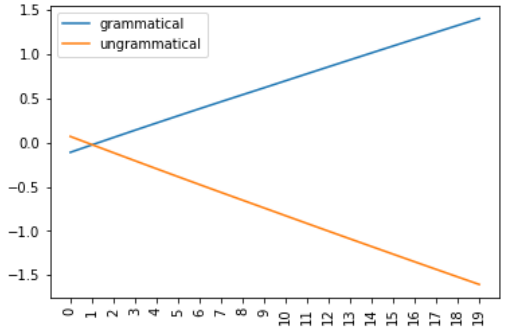
(A) RNN (50)



(B) AbLSTM

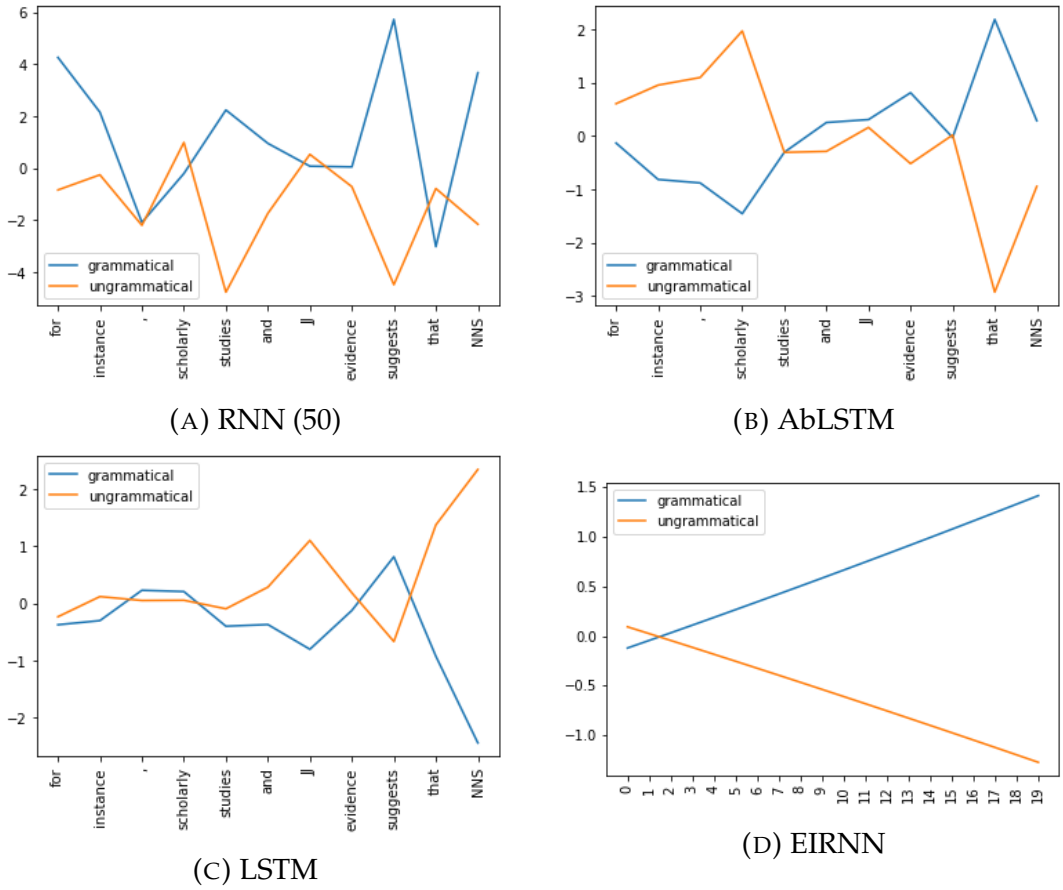


(C) LSTM



(D) EIRNN

FIGURE A.3: Plus 2 : This application of flight tracking is currently in its NN, but is set to grow significantly as **systems get** more connected



(A) RNN (50)

(B) AbLSTM

(C) LSTM

(D) EIRNN

FIGURE A.4: Plus 2 : * For instance, scholarly **studies** and JJ evidence **suggests** that NNS

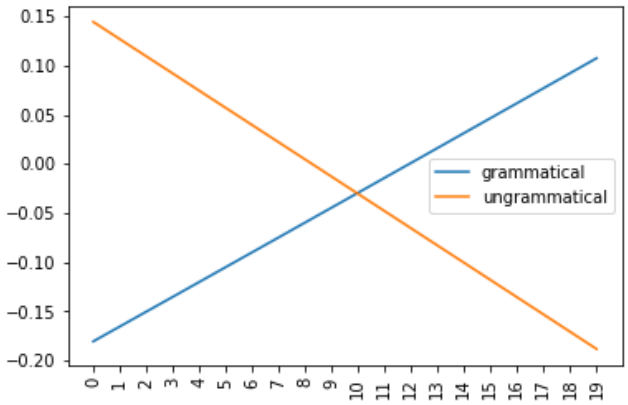


FIGURE A.5: EIRNN - Plus 2 : * The **clusters has** different characteristics

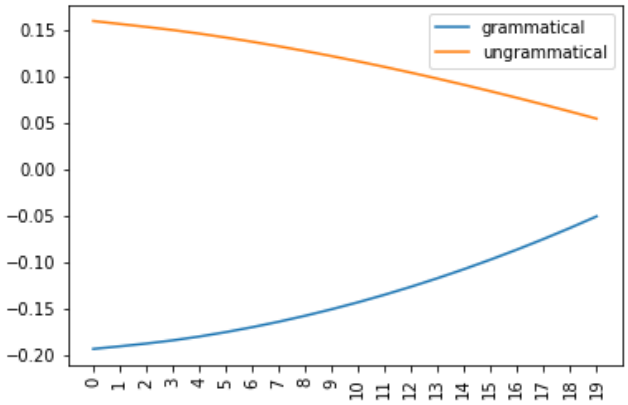


FIGURE A.6: EIRNN - Plus 2 : The film's **cast includes** Anne-Marie Macdonald

Bibliography

Adhiguna Kuncoro Chris Dyer, John Hale Dani Yogatama Stephen Clark 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*, 1426–1436.

H. Francis Song Guangyu R. Yang, Xiao-Jing Wang (2016). "Training Excitatory-Inhibitory Recurrent Neural Networks for Cognitive Tasks: A Simple and Flexible Framework". In: *PLoS Computational Biology*.

Hebb, Donald (2001). "The Organization of Behaviour". In: *Review of Scientific Instruments* 72.12, pp. 4477–4479.

Rosenblatt, F. (1958). "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain". In: *Psychological Review* 65.6.

Socher, Richard (2014). "Recursive Deep Learning for Natural Language Processing and Computer Vision". In: *Ph.D. Thesis, Stanford University*.

T Linzen E Dupoux, Y Goldberg (2016). "Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies". In: *Transactions of the Association for Computational Linguistics 4* Q16-1037, pp. 521–535.