Numerical Sequence Prediction using Bayesian Concept Learning[*]

Damarapati, Mohith (`md4289@nyu.edu`),
Enaganti, Inavamsi Bhaskar (`ibe214@nyu.edu`) and
Rajakumar, Alfred Ajay Aureate (`aar653@nyu.edu`)
Courant Institute of Mathematical Sciences, New York University, New York

Abstract

Patterns are usually generated from a certain set of rules. Mathematical sequences are the type of patterns with rules corresponding to basic mathematical operations like addition and multiplication. Humans learn to predict patterns in flexible ways. When people understand mathematical rules, they can effortlessly generalize those concepts and apply to other numbers. It is also easier for humans to grasp combinations of these primitive rules. Current state-of-the art RNN architectures like LSTMs perform well on temporal data, but they require vast amounts of training examples. Despite vast data, these models struggle to generalize concepts. Our results show that people do well even when the sequences are slightly corrupted. Humans are smart enough to disregard the noise and understand the underlying rule that generated the sequence. We propose a Bayesian model that captures these human-like learning abilities to predict next number in a given sequence.

1. Introduction

A sequence is a regularity with its elements repeating in a predictable manner. Any sequence is built from a certain set of primitives which repeat according to a certain set of rules. Humans are really good at observing these primitives and rules behind the patterns. Given a sequence, people could naturally predict what comes next by learning rules behind it. Current state-of-the-art Recurrent Neural Network architectures like Long Short-Term Memory (LSTM) networks require hundreds or thousands of data to predict the next number in a sequence as they fail to learn richer concepts which humans do very easily [2].

We would be dealing with strictly increasing mathematical sequences in this project. Numbers in a mathematical sequence follow certain rules. In general, a number in a sequence would depend on the number preceding it or its position. (Even if it's dependent on just the position in the sequence, the current number could still be related to its preceding number in some intuitive fashion). But, in cases like Fibonacci series, a number depends on two numbers preceding it. We propose a model that captures human learning abilities for predicting the next number in the sequence using Bayesian Concept Learning.

In general, humans are remarkably adaptable to noisy sequences. People are fairly capable of identifying the underlying rules generating a sequence even when the sequence is corrupted with either stationary noise (where some elements are slightly away from their original hypothesis) or progressive noise (where the errors are propagating across a sequence). Our model also captures this ability of humans to perform well in a noisy environment, meaning identifying the underlying rule and thereby predicting the next number. In this report, we've considered only progressive noise in our Bayesian model while computing likelihood, as stationary noise might be harder to implement using our method. We claim that this type of Bayesian learning approach is similar to humans approach to capturing the rules of the intended sequence.

Over time, people gradually develop their sequence solving skills. At first, they learn to predict sequences which just vary by a constant addition or multiplication factor. As they develop, they could predict sequences with combinations/compositions of multiplication and addition factors. And, when their expertise improves more, they could even predict sequences which depend on two preceding numbers. We are able to capture this gradual learning process of humans in our model with the help of a parameter called **Human Experience Factor -** β_{hef}.

The core ideas and inspiration were from the paper on number game (Ref. [6]) and (Ref. [5]). Although Ref. [4] is somewhat remotely related, our work is pretty much outlined by Ref. [3]. As described in Ref. [5], our models could also be formalised using $lambda$ functions, but we're not using that formalism here.

2. Mathematical Sequence

In a mathematical sequence, usually a number $x(n)$ depends either on just n (or the position) or the previous element(s) in the sequence, $x(n-1), x(n-2), ...$ Here, for simplicity we are going to consider only sequences where each number depends on just the previous number.

Hence, the rule that we consider for generating sequences is: $x(n) = f(x(n-1))$, where $f : \mathbb{N} \to \mathbb{N}$

Here we'll be considering only linear functions as generators or f. So, $x(n) = ax(n-1) + b$, for some positive integers a and b.

Examples of sequences:

Ex. 1.1. Rule: $x(n) = 2x(n-1)$; Example of a sequence generated using this rule: $2, 4, 8, 16, ...$

Ex. 1.2. Rule: $x(n) = 2x(n-1) + 2$; Example of a sequence generated using this rule: $4, 10, 22, 46, ...$

Ex. 1.3. Rule: $x(n) = x(n-1) + 5$; Example of a sequence generated using this rule: $2, 7, 12, 17, ...$

Here, you could notice that Ex 1.1. is a sequence of powers of two.

3. Bayesian Concept Learning: Model Setup

3.1. Primitives and Hypothesis

A lot of strictly increasing mathematical sequences can be generated by a just a series of multiplications ($\times$) and additions ($+$) of certain numbers.

For the sake of simplicity, let us assume just the following 12 primitives, 10 for addition ($+$) and 2 for multiplication ($\times$). Let $A_i(p) = p + i, i \in 1, 2...10$ (meaning add i to the previous number) Similarly, let $M_i(p) = p \times i, i \in 2, 3$ (meaning multiply 2 or 3 to the previous number)

For each of the above defined primitives, we will store a possible transition pair. We assume these pairs as something that humans could infer right way from their elementary math knowledge or memory.

Examples for primitive pair lists:

Ex 2.1. $L(A_1) = \{(1,2), (2,3), ..., (99, 100)\}$
Ex 2.2. $L(M_2) = \{(1,2), (2,4), ..., (49, 98)\}$
Ex 2.3. $L(A_5) = \{(1,6), (2,7), ..., (94, 99)\}$

We generate different hypotheses by combining these primitives - combining one of the M_s's with one of the A_t's to obtain: $x_n = A_t(M_s(x_{n-1})) = s \times x_{n-1} + t$

Here, M_s is from the set of multiplicative primitives $\{M_2, M_3\}$ and A_t is from the set of additive primitives $\{A_1, A_2, ..., A_{10}\}$ So, number of possible such compound hypotheses for a sequence would be $10 \times 2 = 20$.

A compound hypothesis is a combination of primitives (one additive and one multiplicative hypothesis combined). Examples of sequences generated from such compound hypotheses:

Ex. 3.1. H with primitives - $(M_2, A_2) \to x_n = 2x_{n-1} + 2$. Sequence example: $2, 6, 14, 30, ...$ Ex. 3.2. H

with primitives - $(M_3, A_5) \to x_n = 3x_{n-1} + 5$. Sequence example: $5, 10, 15, 20, ...$

3.2. Noise

Before we discuss about the Bayesian method of concept learning let us assume that the training and test sequences could be corrupted by some noise. Here,w.l.o.g. we add standard Gaussian noise $N(0, \sigma)$ (properly normalized so that only integer values are allowed) to the sequence, which could be introduced in the following two ways:

Stationary Noise - Consider a noise free sequence generated from a hypothesis, and now to each term we add noise but with obviously only integer numbers in the sequence.

Example M_2 : $2, 4, 8, 16, 32, 64.$, with noise might become $2, 5, 8, 16, 32, 64.$

Progressive Noise - Here, every time we generate the next term we add noise. Then, we take the term with noise added and continue with the generation.

Example M_2 : $2, 4, 8, 16, 32, 64.$, with noise will become $2, 5, 10, 20, 40, 80$

For the sake of simplicity, we are only considering Progressive Noise in our project. All the models can be effortlessly extended to Stationary Noise too.

3.3. Prior

The framework described here is closely related to the ones described in Ref. [6]. We are assuming that the prior probability would depend on the human experience, meaning, it would be a function of how much familiar the human is to the mathematical sequences that we present here. So, we have the probabilities defined as a function of β_{hef} which varies from 0 to 1 depending on whether the human is completely inexperienced or a math novice! Here we assume that for any β_{hef} or human experience level, the probability of a human being able to identify hypothesis with just one primitive would be the same independent of whether the primitive is A_i or P_i. But, depending on β_{hef} or the human experience level, the probability of a human being able to identify the compound hypothesis (hypothesis formed by both A_i and P_i) corresponding to the given sequence increases for higher values of β_{hef}.

The probability of choosing one of the primitive hypotheses would be, $P(h_{p \, or \, a}) = 1/[(n_p + n_a) + \beta_{hef} n_p n_a]$ Similarly, the probability of choosing one of the compound hypothesis would be $P(h_c) = \beta_{hef}/[(n_p + n_a) + \beta_{hef} n_p n_a]$, where β_{hef} varies from 0 to 1; n_p is the total number of product primitives and n_a is the number of additive primitives and so $n_p n_a$ therefore represents the total number of compound hypotheses that are formed by combining a A_i and a P_i.

3.4. Likelihood

Our aim here is to compute $P(D|h_i)$, where $D = x_1, x_2, x_3, x_4, ..., x_n$ is the given sequence. Here $P(D|h_i)$ is the probability that we get D while using the generating rule h_i.

$P(D|h_i) = P(x_1|h_i) \prod_{j=2}^{n} P(x_j|x_{j-1}, h_i)$

Here, h_i is a combination of primitives. $h_i = \{A_p, M_q\}$, and

$P(x_n|x_{n-1}, h_i) = g(x_n - f(h_i(x_{n-1})))$, where $g(x)$ is a standard normal distribution $N(0, 1)$ and $f(h_i(x(n - 1))) = A_p(M_q(x(n - 1)))$.
$M_q(k)$ is transition of number k in $L(M_q)$.
$A_p(k)$ is transition of number k in $L(A_p)$

(We assume here w.l.o.g., that $P(x_j|h_i) = k$, some constant. We'll see that this constant would get cancelled out in most of our calculations, especially due to Bayesian, and so, it need not be included in our analysis.)

3.5. Posterior

Using Bayes Theorem :
$P(h_i|D) = P(D|h_i)P(h_i)/P(D)$
To find the most likely hypothesis h^* : $h^* = \max_i P(h_i|D)$

After finding h^*, we generate the next number by just applying the hypothesis on the last element of $h^* = \{M_p, A_q\}$.
$x(n + 1) = px(n) + q$

3.6. Human Experience Factor (β_{hef})

We introduce a new parameter to the model which is human experience factor (β_{hef}). The β_{hef} is higher for people who are better at identifying sequences, because of "familiarity". To capture it in the model, we assume that the prior probabilities are a function of β_{hef}, as mentioned in an earlier section. For lower β_{hef} values, prior probabilities for hypothesis with $\{A_1, A_2, ..., A_{10}, M_2, M_3\}$ will be high and for compound hypothesis, it would be low. As β_{hef} increase, probabilities for compound hypothesis also becomes comparable to the primitive hypotheses.

4. Recurrent Neural Networks

We performed experiments specifically on LSTMs. LSTM is a variant of an RNN. They are good at maintaining long term dependencies and hence they are perform well as sequence models. A brief description of a LSTM cell and related equations are explained below. LSTMs are first introduced by S. Hochreiter and J. Schmidhuber in 1991 (Ref.[2]). Gates are the major components of an LSTM which are input gate - $i(t)$, forget gate - $f(t)$, output gate $o(t)$, input modulation gate - $g(t)$ and $c(t)$ - memory cell. At each time step, an LSTM cell takes $x(t)$, $h(t - 1)$ and
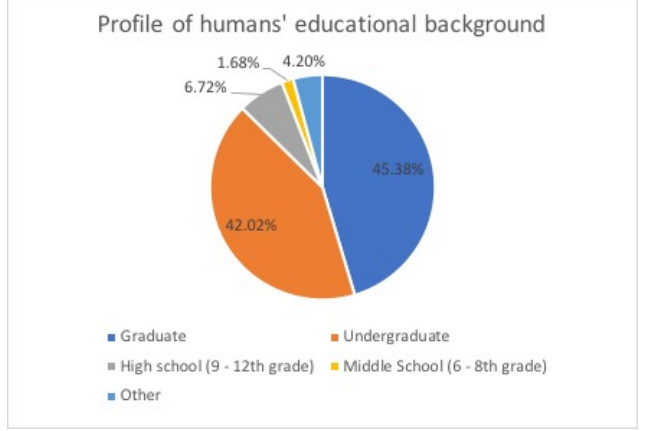

Figure 1. Pie-chart of the distribution of the participants' educational background

$c(t - 1)$ as inputs and outputs $h(t)$ and $c(t)$. Working of an LSTM cell is described using sequence of equations below.

$$
\begin{aligned}
i(t) &= \sigma(W_{xi}x(t) + W_{hi}h(t-1) + b_i) \\
f(t) &= \sigma(W_{xf}x(t) + W_{hf}h(t-1) + b_f) \\
o(t) &= \sigma(W_{xo}x(t) + W_{ho}h(t-1) + b_o) \\
g(t) &= \phi(W_{xc}x(t) + W_{hc}h(t-1) + b_c)
\end{aligned}
\tag{1}
$$

$$
\begin{aligned}
c(t) &= f(t) \cdot c(t-1) + i(t) \cdot g(t) \\
h(t) &= o(t) \cdot \phi(c(t))
\end{aligned}
\tag{2}
$$

Here $\sigma(x) = \frac{1}{1+e^{-x}}$ (sigmoid activation) and $\phi(x) = \tanh x$ ($\tanh$ activation).

Memory gate enables LSTM to learn complex long-term temporal dependencies. Additional depth can be added by stacking these on top of each other.

5. Behavioral Experiments

We conducted an online survey, to observe how humans choose the next possible number among 4 options, more specifically, the 6th number in a sequence where 5 numbers are given. About 119 people participated in the survey from different backgrounds. The profile of the participants is illustrated in Fig. [1]. Since we passed it among our friends and family, we could observe that most of the participants are either graduates or undergraduates. Here, in our analysis, we are ignoring this selection bias and assuming that the participants would still be diverse enough in terms of identifying the underlying rules in a numerical sequence, as that depends more on the cognitive abilities of the humans which are very hard to profile.

The survey had 2 sections of 6 questions each. Each of those questions belong to a particular class/type of sequences. The sequences given in section 1 and their types are mentioned in table - 1. From about 54 participants (who

Sequence Type	Sequence given in section 1
Pure addition	11, 19, 27, 35, 43, ___ a) 86; b) 51; c) 52; d) 53 Ans: b) 51
Addition with noise	5, 7, 9, 12, 14 ___ a) 17; b) 28; c) 12; d) 16 Ans: d) 16
Pure multiplication	5, 10, 20, 40, 80 ___ a) 120; b) 85; c) 160; d) 100 Ans: c) 160
Multiplication with noise	3, 6, 12, 25, 50, ___ a) 75; b) 100; c) 56; d) 150 Ans: b) 100
Pure compound sequence	1, 4, 13, 40, 121, ___ a) 122; b) 364; c) 243; d) 606 Ans: b) 364
Compound sequence with noise	5, 14, 33, 70, 144, ___ a) 580; b) 436; c) 148; d) 292 Ans: d) 292

Table 1. Different class of sequences given to people in the survey. The sequences were given with and without noise or error as can be observed from the sequences in the 2nd column. The options that were given in the survey and the correct option are also given in the same column.

opted to answer more questions), we received responses to 6 more questions of similar types (as the ones in section 1). So, overall we received about 173 responses to each of the 6 type of sequences.

6. Experimental Results

6.1. Results of the online survey

From fig. 2, it is evident that humans are able to detect the generative rules of both noisy and noiseless sequences with similar precision. Humans are able to capture the concepts despite corruption of data with noise. As expected humans with more experience (or education qualification) were able to more easily understand the compound sequences.

6.2. LSTMs

We trained LSTM models separately for additive, multiplicative and compound sequences with and without considering noise. From our results, it is interesting to note that LSTMs are really good at learning addition without noise. Our best LSTM model, with the hyper parameters as mentioned in table - 2, got an accuracy of 99.21% without noise and 70.3% with noise for addition (table - 3). Although LSTMs performed well for addition, they greatly underperformed when compared to humans for the multiplicative and compound hypotheses-based sequences (fig. 5). This
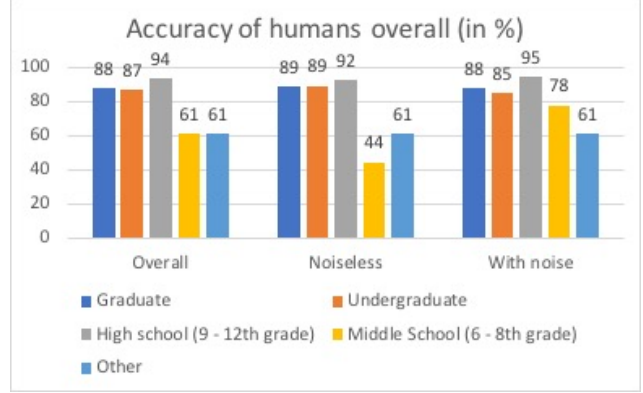

Figure 2. Overall accuracy of humans (left) from different education backgrounds, in predicting the next number in a sequence; the set of bars in the middle and in the right end show the accuracy of the participants in sequence prediction tasks when there is no noise in the sequence and when there is an error/noise in the sequence, respectively.
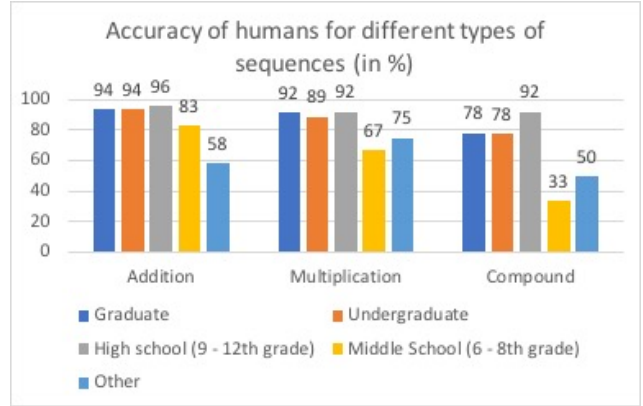

Figure 3. Accuracy of humans from different education backgrounds, while predicting the next number in a sequence. The set of bars to the left corresponds to sequences that are generated using just addition operation; similarly the set of bars in the middle corresponds to sequences that are generated using just multiplication operation and the set of bars in the right corresponds to sequences that are generated using both multiplication and addition operations.

brought down the overall accuracy of LSTMs and its accuracy for sequences with and without noise too (fig. 4).

Data - Deciding right amount of data for a fair training of LSTM was a challenge for us. We considered 999 as the maximum number for generating addition hypotheses to get a good amount of data to train our LSTM model. But for the multiplication and combination hypotheses, 999 is a very low number and we could barely get sequences in that range. Hence, we considered 99999 as our maximum number for multiplication and combination hypotheses. Another interesting aspect we considered for ensuring fair training of LSTM is to select a certain number of sequences such that
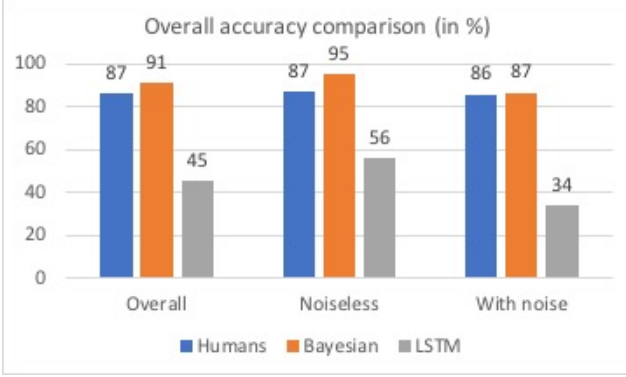
Figure 4. Comparing the overall accuracy (left) of humans with the overall accuracy of the Bayesian model and LSTM, in predicting the next number in a sequence. The set of bars in the middle and in the right end show the comparison of the accuracy of humans with that of the Bayesian model and LSTM for sequences that does not have errors and the noisy sequences (or the sequences with error), respectively.
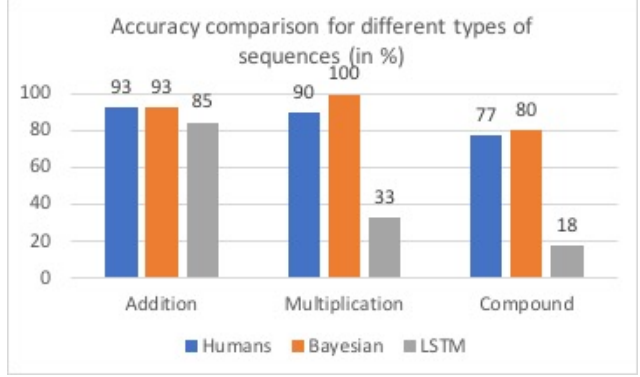

Figure 5. Comparing the accuracy of humans from different education backgrounds, with the accuracy of the Bayesian model and LSTM, while predicting the next number in a sequence. The set of bars to the left compares the accuracy of humans to that of the Bayesian model and LSTM for sequences that are generated using just addition operation; similarly the set of bars in the middle corresponds to sequences that are generated using just multiplication operation and the set of bars in the right corresponds to sequences that are generated using both multiplication and addition operations.

Hyper parameters	Addition	Multiplication	Compound
Epochs	30	30	30
Batch size	16	16	16
No. of hidden layers	2	2	2
Train data	2500	500	300
MAX_number	999	99999	99999

Table 2. Hyper parameters used in the LSTM implementation for different class of sequences.

probability of seeing all pairs is at least 70. We considered this to be a reasonable number for investigating LSTMs on our tasks. More specifically, we chose 2500 sequences out of 10000 possible for training addition hypotheses, 500 out of 1700 possible for multiplication hypotheses and 300 out of 1000 possible for combination hypotheses. Despite giving the LSTM a slight advantage, their performance is still low for multiplication and combination hypotheses (fig. 2 and table - 3).

Data Encoding - From our experiments, we observed that encoding of the data fed to LSTMs is an important factor while training LSTM. We first followed one-hot encoding for each number in our dataset. Results obtained were really bad and it was totally unfair for LSTMs to be trained that way. We then considered one-hot encoding for each digit of a number which is introduced in Ref. [1]. So, size of our encoding now became 30 for addition hypotheses (as maximum number is 999) and 50 for multiplication and combination hypotheses (as maximum number is 99999).

Hyper Parameters - We considered a batch size of 16 with 2 hidden layers and trained all our models for 30 Epochs. We followed 80-20 train test split (table - 2).

6.3. Bayesian Model

Despite noise we noticed that the Bayesian model was able to very accurately predict the generating rule/hypothesis.

Here is an example. Consider the sequence : $1, 2, 3, 5, 8, ...$ There are two highly probable rules that could have generated this sequence namely (M_2, A_1) with no noise or (M_2) with noise in each term.

Below are the results with the following parameters: Noise mean = 0

Noise variance at each element = 0.66 (This ensures half of the sequences are noiseless)
Prior of A, M, C : $(1, 1, 1)$ or $\beta_{hef} = 1$ (meaning all the hypotheses including the compound hypotheses are equal)
Result:

$$
\begin{aligned}
&A_2 : 0.74162549807248, \text{ Prediction} :10 \\
&A_1 : 0.19419344602990238, \text{ Prediction} :9 \\
&M_2 : 0.0639090892641122, \text{ Prediction} :16 \\
&A_3 : 0.0002719666030357511, \text{ Prediction} :11
\end{aligned}
\tag{3}
$$

Here, 'Prediction' refers to the next possible number predicted by the model.

6.4. Impact of Human Experience Factor

From the results, it can be seen that humans find it harder to solve compound sequences. We noticed that experienced people were able to identify these compound sequences easily. An expert tends to add more weight to the prior associ-

Sequence Type	Humans	Bayesian Model	LSTM
Pure addition	94.22	99.99	99.21
Addition with noise	91.33	74.80	70.20
Pure multiplication	91.33	99.99	48.00
Multiplication with noise	88.44	99.99	18.00
Compound sequence	76.88	56.42	21.00
Compound sequence with noise	78.03	89.11	15.00

Table 3. Comparing accuracy of humans (derived from the behavioral experiment) with that of the Bayesian model and LSTM for the 6 class/type of sequences. The accuracy mentioned here are in %.
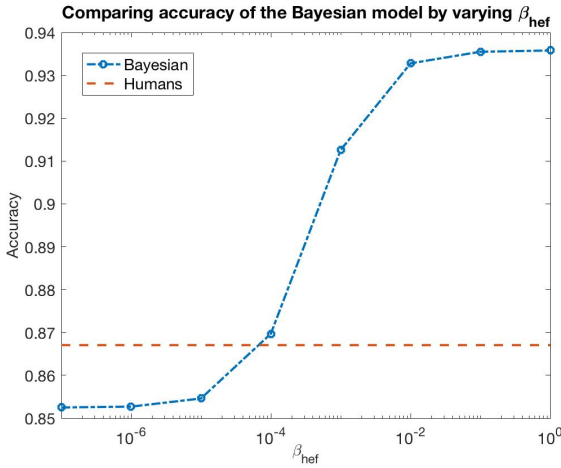

Figure 6. Comparing the accuracy of the Bayesian model with human accuracy for different values of β_{hef}. The x-axis is scaled logarithmically to illustrate the variation in accuracy while changing the β_{hef} factor across orders of magnitude.

ated with the compound sequence. We were able to capture this scenario using Human Experience Factor (β_{hef}). In particular, we noticed that graduates were more comfortable with compound sequences compared to high school students in general. This could possibly be a result of priors developing with experience.

On another note, we tried to approximate the β_{hef} factor averaging over a hundred humans as a group by comparing with our Bayesian learning model. This plot is shown in Fig. [6]. The red line at 0.86705 refers to the average human's accuracy, averaged over hundred participants. Whereas the blue line refers to the Bayesian model's accuracy for different values of β_{hef}^{eff}. The point at which both the plots meet would correspond to the effective β_{hef}^{eff} value for the entire group of participants as per the Bayesian model. It turns out be about $\beta_{hef}^{eff} = 0.0001$. At that value of $\beta_{hef} = 0.0001$, the overall accuracy of the Bayesian model matches with that of the group of human participants.

7. Conclusion

We noticed that LSTMs performed exceptionally well on noiseless addition and reasonably well on its noisy counterpart. Despite LSTMs being able to grasp the additive rule, it failed to understand multiplication and compound sequences which are just meta-additive rules.

As was expected, humans were able to figure out addition and multiplication sequences accurately and compound sequences to a reasonable extent.

The Bayesian learning model performed a lot better compared to LSTMs, specifically the multiplicative and compound sequences. Despite the noise, the Bayesian model was able to accurately predict the generative rule for a sequence.

From our behavioral data, we were able to gauge certain trends of human experience with sequences. An higher Human Experience Factor (β_{hef}) implied increased familiarity with compound sequences. From this we conclude that as humans come across more sequences, their priors also develop in such a way that they would be able to solve compound sequences. This is analogous to how children improve their priors as they learn more patterns. From our analyses, we observed that this kind of sequential concept learning does not seem to be captured by LSTMs.

8. Future Work

We have to admit that our survey results were skewed due to lack of proportionate representation of different groups. We plan to rectify this with proper control groups. More steps have to be taken to compensate the inherent selection bias in a free-for-all survey!

Our Bayesian model can be improved by using additional psychological and cognitive factors that might be affecting the way humans identify patterns in general. These may be derived from human data. It could be factors like, giving more weight to the latter numbers in the sequence, accounting for familiarity with round numbers ($+1, +2, +5, +10$) and inherent grouping of numbers in the sequence that were not a result of our hypotheses. For example: the sequence $3, 5, 7, ...$ could either be identified as a sequence of odd numbers or a sequence of prime numbers).

We could also extend our model to more complex hypotheses, with very little modification. For instance, we could extend our model to Fibonacci-like sequences which depend on the previous two numbers unlike our hypotheses which only depended on the previous number.

Acknowledgments

Kagitha from NIT Surat (India) for the useful discussions and suggestions.

References

[1] A. Graves. Adaptive computation time for recurrent neural networks. *CoRR*, abs/1603.08983(1):1 – 19, 2016.

[2] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 1997.

[3] M. Hutter. How to predict (sequences) with Bayes, MDL, and Experts. *http://www.hutter1.net/ai/predict.htm*, pages 1–90, 2005.

[4] P. C. E. L. L. M. K. R. M. S. G. T. J. T. P. Jaini, Z. Chen and P. Poupart. Online bayesian transfer learning for sequential data modeling. *International Conference on Learning Representations (ICLR)*, 2017.

[5] N. D. G. Steven T. Piantadosi, Joshua B. Tenenbaum. Bootstrapping in a language of thought: A formal model of numerical concept learning. *Cognition*, 123(2):199–217, 2012.

[6] J. B. Tenenbaum. Rules and similarity in concept learning. *Advances in Neural Information Processing Systems 12*, pages 59–65, 2000.