IF OR IF NOT: CAN NEURAL NETWORKS UNDERSTAND LOGICAL ENTAILMENT?

Jasmin Omanovic*
Department of Mathematics
University of Western Ontario
London, ON, Canada
jomanovi@uwo.ca

March 29, 2019

ABSTRACT

In this abstract we review and analyze some of the concepts discussed in "Can Neural Networks Understand Logical Entailment" by Richard Evans, David Saxton, David Amos, Pushmeet Kohli and Edward Grefenstette. In particular, we study the robustness of the data set generated by the authors along with implementing several baseline and benchmark models in an effort to reaffirm the results attained in the paper. Lastly, we offer an interpretation of the PossibleWorldsNet which sheds some light on the implementation put forth by the authors.

1 Introduction

The modern success of Neural Networks at achieving remarkable results in a variety of tasks no longer needs to be cited, it is common place. In many cases, or as "No Free Lunch" [3] would suggest, in all cases the success of a particular neural architecture X at solving a problem Y is a consequence of the inductive bias inherent to the particular architecture X. As a consequence, in order to understand what a neural architecture is good at learning we should look toward studying what kind of problems it's good at answering. In "Can Neural Networks Understand Logical Entailment" Evans et al do just that and suggest a good starting place for such an inquiry is that of logical entailment which is inherently a structural task governed only by connective rules.

2 Methodology

As a measure of a neural architectures ability to infer, encode and relate syntactic sequential data Evans, et al generated a robust data set $\mathcal{D}$ by sampling from a distribution of triplets *(A,B,E)* where *A* and *B* are logical formulae and *E* is an indicator function conditioned on whether or not A entails B i.e. (x$\vee$y, x, 0), (x$\wedge$y, x, 1) $\in \mathcal{D}$. The samples *(A,B,E)* are drawn under constraint from a larger distribution to minimize superficial differences in formula belonging to positive and negative examples. This constraint is necessary as simply drawing from a randomly generated sample of formula $\mathcal{R}$ such that $|\mathcal{R}| = n$ we would find that $P(A \models B) \longrightarrow 0$ as $n \longrightarrow \infty$ for $A, B \in \mathcal{R}$. This is a consequence of the semantic (model-theoretic) definition of entailment:

$$A \models B \text{ iff for every truth-table row w : } A(w) \text{ is true implies } B(w) \text{ is true.} \tag{1}$$

Note that problems such as logical entailment, which in the sense above can be rephrased in terms of satisfiability tend to be NP-hard or worse [6]. Thus to evaluate entailment of a sentence with *n* propositional variables would require a worst-case of $\mathcal{O}(2^n)$ operations although efficient searches with back-tracking, unit propagation and adaptive branching have been successful in improving computation time on average (one such successful algorithm is minisat, see [4]). An interesting question, which was not addressed in the paper is to what extent data generation via back-chaining (using the rules of inference) of formula suffers from exploitability.

*Implementations can be found at https://github.com/AskingQ

2.1 Experiments

The results obtained by these experiments are my own and should not be seen as contradicting those achieved by Evans et al in [1]. All models mentioned in this paper were implemented in Tensorflow 2.0 and run on a Intel(R) Core(TM) i7-4710HQ CPU @ 2.50GHz with GeForce GTX 860M . We did not apply the data augmentation technique (permuting propositional symbols) suggested by the authors opting instead to study the models internal ability to differentiate between different symbols.

2.1.1 Baselines

To test the robustness of the data set against artifacts in the data generation process we tested three different neural architectures: Linear BoW, MLP BoW, MLP TF-IDF. The central component which differentiates these architectures is the encoding scheme. In BoW we represent formulae by one-hot encoding symbol frequency for every unique symbol (word) in a formula while in TF-IDF formulae are encoded via symbol (term) frequency adjusted by a weighting factor which accounts for uniqueness across other formula (documents). We achieve results which are comparable to those of Evans et al on Linear BoW and MLP BoW. Although the performance of these models is better than random guessing (suggesting some artificial aspects of the data generation process remain) it is by and large expected given that the data set is balanced and the various encoding schema capture topical information at the expense of discarding vital structural/syntactical information.

2.2 Benchmarks

We studied two classes of benchmark models tested by Evans et al. The first of these classes are Encoders which attempt to first represent (encode) each logical expressions into an appropriate vector space after which an MLP is applied in order to determine logical entailment (similarity). We implemented several models of this class including ConvNet Encoders [5], LSTM Encoder, BiDirLSTM Encoder [7]. In implementing ConvNet encoders, it was necessary to explicitly pre-compute possible layer configurations when Grid searching through the hyper-parameters suggested by the authors. Our specific implementation was plagued by vanishing/exploding gradients in almost all configurations and failed to outperform MLP BoW in general. For LSTM and BiDirLSTM Encoders we achieved results similar to those of Evan et al on both the LSTM Encoder and BiDirLSTM. It is important to note however that the implementation which achieved the highest accuracy on validation data differed from Evans et al in that it applied a time distributed (to every cell output) MLP to each sequential output of the LSTM.

The second class of benchmark models we investigated were relational. This means that we no longer considered the meaning of two formulae as distinct from one another in terms of entailment. Indeed, relational models such as LSTM Traversal and BidirLSTM Traversal represent the entirety of the logical expression $A \models B$ and traverse the expression as a many-to-one classification task. Similarly to the ConvNet Encoder our implementation was plagued by vanishing gradients with our model adapting strategies consistent with guessing the majority class. We saw moderate improvements applying a time-distributed Dense layer to all sequential outputs and stacking BidirLSTM cell outputs, we could not however achieve results similar to the authors due to memory constraints.

2.3 PossibleWorldsNet

Based on the model theoretic notion of entailment (see eq (1)), Evans et al. defined a continuous approximation to entailment via a series of refinements approximating logical connectives. The essential idea here is that entailment can be treated as a semantic notion predicated on a set of possible worlds $\mathcal{W}$. Note that typically, these worlds are thought of boolean vectors assigning a propositional variable to T or F encapsulating all possible truth combinations, however in the continuous analog these vectors need simply have real value coefficients and need not encapsulate all truth configurations. To evaluate these worlds the authors propose a variant of TreeNets [2][8][9][10] called PossibleWorldsNet [1] which encodes the syntactic parse of each logical formula respectively. For example, to determine $x \vee y \models x$ we must first encode $x \vee y$ and x separately. To encode the former, we apply a pre-defined parser (which is given to the model rather than learned) turning $x \vee y$ into $(\vee, (x, y))$. Having done so, we evaluate the propositional variables x, y in their respective worlds (this can be thought of assigning a truth value to each variable) encoding relevant information in each propositional variable separately i.e. $W^x w$ and $W^y w$ where W^x, W^y are learnable weight matrices and $w \in \mathcal{W}$ is the particular world we are evaluating. Evaluating $\vee$ we apply a linear layer specific to $\vee$ on the the concatenation of our representations. Applying this process repeatedly, we encode each side of the expression $x \vee y \models x$ into a vector and apply a linear layer (corresponding to $\models$) to the concatenation of the encoded vectors. A significant issue in our implementation is that it consistently failed to progress past the training stage due to time/memory issues across all levels, reducing the number of worlds required more epochs and too much time while parallelizing the computation by batching operations across several sentences took too much memory.

Table 1: Baseline models

Model	Omanovic	Evan et al.
Linear BoW	52.1	52.6
MLP BoW	54.6	57.8
MLP TF-IDF	52.3	-
ConvNet Encoder	51.8	59.3
LSTM Encoder	62.6	68.3
BidirLSTM Encoder	60.5	66.6
LSTM Traversal	53.4	62.5
BiDirLSTM Traversal	54.5	63.3
PossibleWorldsNet	-	98.7

References

[1] Richard Evans, David Saxon, David Amos, Pushmeet Kohli, Edward Grefenstette. Can Neural Networks Understand Logical Entailment *6th International Conference on Learning Representations* (ICLR), 2018

[2] Miltiadis Allamanis, Pankajan Chanthirasegaran, Pushmeet Kohli, Charles Sutton. Learning Continuous Semantic Representations of Symbolic Expressions *https://arxiv.org/pdf/1611.01423.pdf*

[3] David H. Wolpert and William G. Macready. No Free Lunch Theorems for Optimization *Transactions On Evolutionary Computation*, (IEEE) VOL. 1, NO. 1, APRIL 1997 67

[4] Niklas Sorensson and Niklas Een. Minisat v1. 13-a sat solver with conflict-clause minimization. SAT, 2005(53):1–2, 2005.

[5] Xiang Zhang, Junbo Zhao, Yann LeCun. Character-level convolutional networks for text classification *NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems* Volume 1 Pages 649-657.

[6] Garey, Michael R.; David S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness* W. H. Freeman Co. New York, NY, USA ©1990

[7] Hochreiter, S. Schmidhuber, J. Long short-term memory. *Neural Comput.* 9, 1735–1780 (1997)

[8] Kai Sheng Tai, Richard Socher, and Christopher D Manning. *Improved semantic representations from tree-structured long short-term memory networks.* arXiv preprint arXiv:1503.00075, 2015.

[9] Phong Le and Willem Zuidema. *Compositional distributional semantics with long short term memory.* arXiv preprint arXiv:1503.02510, 2015.

[10] Xiaodan Zhu, Parinaz Sobihani, and Hongyu Guo. *Long short-term memory over recursive structures.* In International Conference on Machine Learning, pp. 1604–1612, 2015.