CSE 505 - Computing with Logic
Project - Phase 3
DeepMath - Deep Sequence Models for Premise Selection

Mihir Chakradeo - 111462188

December 21, 2017

1 Introduction

Automated theorem proving (also known as ATP or automated deduction) is a sub field of automated reasoning and mathematical logic dealing with proving mathematical theorems by computer programs [4]. These theorem provers have a large corpus of already proven theorems written in form of First Order Logic (FOL) or in Higher Order Logic (HOL). The query is also a mathematical statement to be proven, in form of FOL or HOL. The provers then perform searching on the corpus of already proven theorems to find lemmas or statements which can be used in proving the query. This approach is a bruteforce approach. The available corpus can be as large as 56k theorems (Mizar Mathematical Library). In such a huge search space, bruteforce strategy is really inefficient. So, ATP's started using search pruning techniques such as heuristics or machine learning algorithms such as K-Nearest Neighbors (KNN). However, these approaches do not guarantee that a proof can be found. Moreover, they are not close to what we get from "human intuition".

This paper proposes a new technique for improving ATP's by introducing "human intuition" using Deep Learning. The paper proposes using deep learning for premise selection from the available corpus of theorems, that is, selection of statements from the corpus that will most likely lead to an automatically constructed proof of the query by the ATP.
The paper uses the following approach:

- Given a mathematical statement S in FOL/HOL
- Given a large number of already proven theorems (Mizar Library [6])
- Selecting subset of proven statements that are most useful for proving S (Premise selection)

- Pass it to the ATP (E Prover), dramatically reducing the search space

After going over the paper and the git repo [3], I decided to try to find how the models will work for a similar set of problems in Higher Order Logic.

2 Background

The author's of DeepMath have also worked on premise selection for Higher Order Logic. They have this dataset called holstep, which is basically a dataset for higher order logic theorem proving [5]. They have published baselines on what very basic techniques of machine learning are able to achieve on that dataset, which can be found here [2].

2.1 Models which they have implemented

1. Unconditioned Logistic Regression
   This is the worst model used, whose motivation is to give a baseline on what these models can achieve
2. Unconditional 1D CNN
   The motivation of using this model is to figure out the local patterns amongst the tokens
3. Unconditional 1D CNN with LSTM
   The motivation of using this model is to figure out the importance of order in the feature sequence

Here, the unconditioned keyword means that only the step in a theorem to be proven was given to the model, that is, no other context is given.
Similarly, they also tried out models with Conditioning on them, that is, along with the statement of the theorem, they also provided the conjecture statement as a context.

1. Conditioned Logistic Regression
2. Conditional 1D CNN
3. Conditional 1D CNN with LSTM

2.2 Finding's of the paper

1. Unconditioned Models

   - Logistic Regression - achieved an accuracy of 71%
   - 1D CNN - achieved an accuracy of 82%
   - 1D CNN-LST - achieved an accuracy of 83%

2. Conditioned Models

   - Logistic Regression - achieved an accuracy of 71%
   - 1D CNN - achieved an accuracy of 81%
   - 1D CNN-LST - achieved an accuracy of 83%

These results were obtained with following hypeparameters:

- Epochs: 40
- Batch Size: 64
- Max length of sentence: 512
- Samples per epoch: 128000

The observation of the paper was that the conditioned models gave almost the same accuracy as that of the unconditioned models. That is, the models are not able to leverage information provided by the context of the conjecture. The only improvement the conditioned models provided were faster convergence. The conclusion of the paper was that despite of using weak models, they were still able to predict statement usefulness with a high accuracy.

2.3 My Goals

My goal was to verify the findings of the paper by applying other sequence to sequence models such as:

- 1D CNN-RNN (Unconditioned and Conditioned)
- 1D CNN-GRU (Unconditioned and Conditioned)
- 1D CNN-Encoder Decoder (Unconditioned and Conditioned)

3 My Contributions

Following subsections will state what I implemented as part of the phase 3 and the corresponding findings.

1. **1D CNN with RNN**

   - Unconditioned 1D CNN-RNN In this module, I implemented a model which first creates the word embeddings, then passes them on to a Convolutional Neural Network (CNN) with two convolution layers and max pooling. The result is then passed on to a Recurrent Neural Net (RNN) (because the data is a sequence). Finally, the output is fed to a binary classifier (Logistic Regression). The following figure shows the architecture:

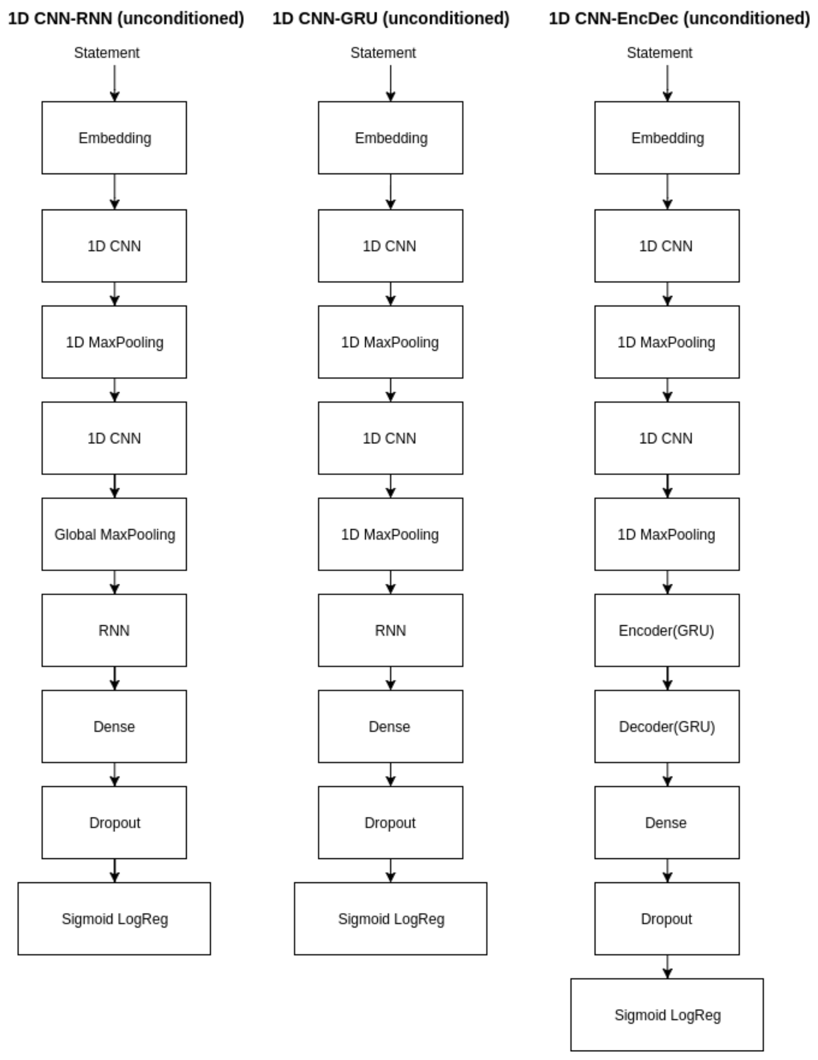

Figure 1: Unconditioned Models

- Conditioned 1D CNN-RNN Same architecture as in unconditioned, with the difference that context of the conjecture is provided, that is, statements from the conjecture to be proven are trained on the same model simultaneously with the subset of statement chosen, and then are combined and then are given to the dense layer. Figure 2 shows architecture of all Conditioned models.

2. **1D CNN with GRU**

   - Unconditioned 1D CNN-GRU In this module, I implemented a model which first creates the word embeddings, then passes them on to a Convolutional Neural Network (CNN) with two convolution layers and max pooling. The result is then passed on to a Gated Recurrent Unit (GRU) (because the data is a sequence). Finally, the output is fed to a binary classifier (Logistic Regression). Figure 1 shows the architecture.
   - Conditioned 1D CNN-GRU Same architecture as in unconditioned, with the difference that context of the conjecture is provided, that is, statements from the conjecture to be proven are trained on the same model simultaneously with the subset of statement chosen, and then are combined and then are given to the dense layer. Figure 2 shows architecture of all Conditioned models.

3. **1D CNN with Encoder Decoder**

   - Unconditioned 1D CNN-Encoder Decoder In this module, I implemented a model which first creates the word embeddings, then passes them on to a Convolutional Neural Network (CNN) with two convolution layers and max pooling. The result is then passed on to a Encoder (GRU) then to a Decoder (because the data is a sequence). Finally, the output is fed to a binary classifier (Logistic Regression). Figure 1 shows the architecture.
   - Conditioned 1D CNN-Encoder Decoder Same architecture as in unconditioned, with the difference that context of the conjecture is provided, that is, statements from the conjecture to be proven are trained on the same model simultaneously with the subset of statement chosen, and then are combined and then are given to the dense layer. Figure 2 shows architecture of all Conditioned models.

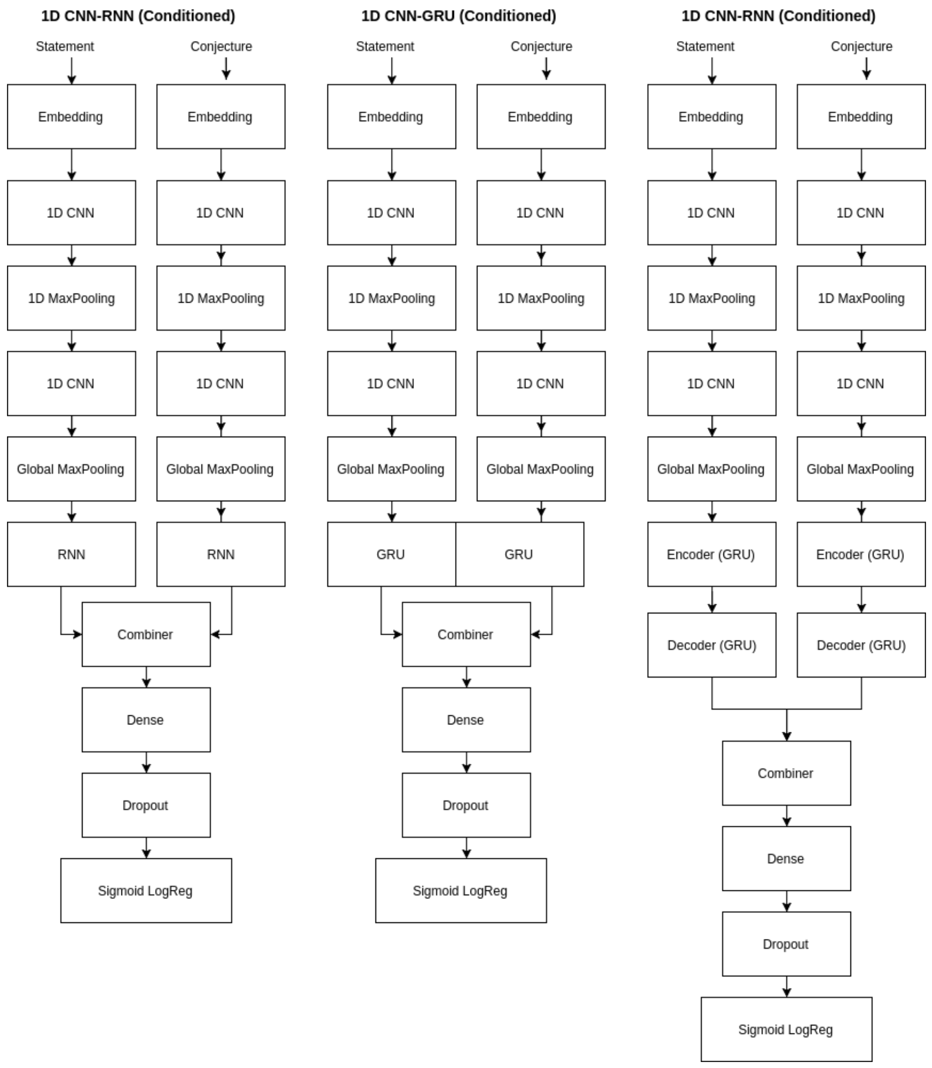

Figure 2: Conditioned Models

4 Findings

I trained and tested the models mentioned above using the following hyperparameters:

- Epochs: 2
- Batch Size: 64
- Max length of sentence: 512
- Samples per epoch: 200

The accuracy came very low compared to that achieved by the paper, this is simply because of the difference in number of epochs (2 vs 40).

id	**Model**	**Accuracy (%)**	**Loss (%)**
1	Unconditioned 1D CNN-RNN	65	62.33
2	Conditioned 1D CNN-RNN	65.45	62.66
3	Unconditioned 1D CNN-GRU	63.8	63.48
4	Conditioned 1D CNN-GRU	66.5	61.57
5	Unconditioned 1D CNN-Encoder Decoder	64.74	62.7
6	Conditioned 1D CNN-Encoder Decoder	67.1	60.7

It can be seen from the table that there is not much difference in the accuracy provided by Conditional vs Uncondiional models.

5 Conclusion

The accuracies of the conditional and unconditional models do not vary much, which means that the models are not able to harness the structure of the theorem to find good features. This was what the paper had concluded. Thus, the models which I implemented are consistent with the paper's findings.

6 Testing Environment and other information

- Language Used: Python
- Libraries Used: tensorflow, keras, numpy
- Test environment: Ubuntu 16.04.3 LTS, Linux 4.10.0-37-generic x86_64

Project Link: https://github.com/mihirchakradeo/CSE505-Project

References

[1] Alex A. Alemi, Francois Chollet, Niklas Een, Geoffrey Irving, Christian Szegedy, Josef Urban *DeepMath - Deep Sequence Models for Premise Selection* 2016.

[2] Cezary Kaliszyk, Francois Chollet, Christian Szegedy *HOLSTEP: A MACHINE LEARNING DATASET FOR HIGHER-ORDER LOGIC THEOREM PROVING* ILP 2017.

[3] DeepMath Github Repo retrieved from URL
$https://github.com/tensorflow/deepmath$

[4] Wikipedia *Automated theorem proving* retrieved from URL
$https://en.wikipedia.org/wiki/Automated_theorem_proving$
$https://github.com/girving/deepmath/tree/master/holstep_baselines$

[5] HOL Step *Machine Learning Dataset for Higher-Order Logic Theorem Proving* retrieved from URL $http://cl-informatik.uibk.ac.at/cek/holstep/$

[6] Mizar Mathematical Library *Formalizing 100 Theorems in Mizar* retrieved from URL $http://mizar.org/library/$