COMP551 Mini-Project 4: Reproducibility in Machine Learning Write-Up

Joyce Liu
McGill University
joyce.liu2@mail.mcgill.ca
260989135

Yoah Qiu
McGill University
yuhao.qiu@mail.mcgill.ca
260985044

Mihail Calitoiu
McGill University
mihail.calitoiu@mail.mcgill.ca
260972537

Reproducibility Summary

We have chosen to reproduce the Deep Neural Decision Trees by Yongxin Yang, Irene Garcia Morillo, and Timothy M. Hospedales. [1]

Scope of Reproducibility

We examine the reproducibility of the Deep Neural Decision Trees (DNDT) comparison results to traditional Decision Trees (DT) and Neural Networks (NN). We reproduce the results seen in Table 1 and Table 2 of the paper and verify the claim that, although no clear advantage was found among the three models, DT performs the best, followed by DNDT and then lastly NN.

Methodology

Given that the official GitHub repository implements cut-points, a concept that we will not reproduce, we reimplemented the model and experiments for DNDT. Additionally for DT and NN, we independently developed the models using the `sklearn` library.

Results

Our research partially replicated the general result of referenced paper, specifically the comparative accuracy of DT, DNDT and NN. Considering accuracies within 1% as ties, we assert that the winning models (for the 10 datasets) are consistent for our experiment and the paper.

When contrasting our findings to the paper, the rankings between the losing models (including the margin of superiority) have high variability for the Car Evaluation, Poker Hand, and Image Segmentation datasets. The highest difference is for Image Segmentation, where our implementation of NN scored 2nd place with accuracy of 92.2% compared to the paper's finding of 3rd place with accuracy of 47.5%. This is a difference of 44.7%, which is significant. Although this variation is not critical for the investigated claim, it does show that some variation exists in model parameters and/or dataset characteristics between our experiments and the paper. We are unable to reproduce the paper's claim that DNDT performs better than NN; our results show that NN performs better, with one additional win. However, it is important to note that the paper investigates three additional datasets, where DNDT has the highest accuracy for two. Considering these additional wins and the mean reciprocal errors, DNDT might perform better than NN. However, we cannot definitively confirm this.

What was easy

It was easy to implement the code; the structures used were not overly complex. Additionally, the paper is written clearly and easy to follow.

Submitted to ML Reproducibility Challenge Assignment for Comp 551 McGill University. Do not distribute.

What was difficult

While we cannot say that it was 'difficult' per say, we encountered certain challenges and inconveniences that required additional attention. Firstly, we were unable to find some of the datasets. Secondly, when implementing the code for DT and NN, the paper abstained to write down some of hyper-parameters they used to train the model. Thirdly, the code in their repository was commented very little, making the understanding of the inner workings very difficult.

Communication with original authors

None

1 Introduction

In this mini-project, we reproduce the results from Deep Neural Decision Trees[1], and compare the results of the ML structures on datasets, shown in Table 1 and 2. More specifically, we train the Deep Neural Decision Tree (DNDT), the traditional Decision Tree (DT) and the Neural Network (NN) on 'Iris', 'Haberman's Survival', 'Car Evaluation', 'Breast Cancer Wisconsin', 'Pima Indians Diabetes', 'Poker Hand', 'German Credit', 'Connect-4', 'Image Segmentation', and 'Covertype' datasets. We find that, due to the lack of some specified hyper-parameters, the results of some models do not align with the performance noted in the paper.

2 Scope of reproducibility

We aim to verify the following claim:

- Decision Trees is the overall best performing model out of the three, followed by Deep Neural Decision Tree, and then Neural Network.

3 Methodology

While there was code written in the authors' official GitHub, the repository only provides code for a specific demo run on a preprocessed 'Iris' dataset and takes in consideration the concept of cut-points; instead, we chose to re-implement the model and experiments. As a side note, the reason why we are discarding the concept of cut-points is because for this set of experiments, the cut-points parameter is set to the same everywhere for the following experiments. Our DNDT creates a decision tree classifier from the intermediate layers of network. As for the other models, we have written ourselves as there was no code at all in the repository. The code for DT and NN models used the `sklearn.tree`, and the `tensorflow.keras` library, and all experiments are written from scratch. No special GPUs were used.

3.1 Model descriptions

Deep Neural Decision Tree (DNDT):

Type: The authors behind this paper proposed a neural network with special architecture; it is a model that maintains the performance of a traditional neural network, while allowing interpretability, through the decision tree structure.

Number of parameters: 4

No pretraining

Decision Tree (DT):

Type: Traditional Decision Tree

Number of (specified) parameters: 2 (criterion as 'gini' and splitter as 'best')

No pretraining

Neural Network (NN):

Type: Traditional Neural Network

Number of (specified) parameters: 2 (# hidden layers as '2' and # neurons as '50')

No pretraining

3.2 Datasets

All datasets used in our experiments can be summarized in Table 1, seen below.

	# instances	# features	# classes	splits	preprocessing
Iris	150	4	3	-	-
Haberman's Survival	306	3	2	-	transform into `.csv`
Car Evaluation	1728	6	4	-	encode categorical data
Breast Cancer Wisconsin	699	9	2	-	encode categorical data
Pima Indians Diabetes	768	8	2	-	transform into `.csv`
Poker Hand	1025010	10	10	97.5% test	transform into `.csv` & undo split
German Credit	1000	20	2	-	encode categorical data
Connect-4	67556	42	3	-	transform into `.csv` & encode categorical data
Image Segmentation	2310	19	7	90.90% test	transform into `.csv` & undo split
Covertype	581011	54	7	-	transform into `.csv`

Table 1: Collection of datasets from UCI and Kaggle

3.3 Hyperparameters

Excluding the few hyper-parameters defined in the paper, there is little information regarding model parameters and it would be out of scope to test out all possible combinations. Therefore, we simply assumed default settings to start off. In the same logic, since the training hyper-parameters were not given either, we let them be the default (e.g.: for `train_test_split()`, we let the split be default 0.25, for NN `Sequential`, the activation function is a simple ReLu, etc.). Seeing as the results obtained are very close to the original, this is satisfactory.

Furthermore, as the only parameter given in the paper for DNDT was the cut-points, which we deemed unnecessary, and the fact that the Adam optimizer was used in one of their demo code files, we took the liberty to choose the rest of the parameters using a manual search. In the end, the closest matching hyperparameters are what we currently have in `DNDT.ipynb`. Another interesting result was 2 hidden layers and 50 neurons each, following the NN structure, performs poorly and deviates from the accuracies by up to 30%.

3.4 Experimental setup and code

Each model has its own file, and for each file, the datasets are run on in the same order to make comparison between them easier. The structure of the files is as follows: imports first, followed by the model, then experiments on the datasets listed previously.

Following the paper's methodology, which uses accuracy to compare models, we also used accuracy to compare the models. This also helps with verifying whether our experiments exactly replicate the paper's experiments.

3.5 Computational requirements

No special GPU was used. Although running on CPU is known to be a slower than the T4 GPU provided by Google Colab, it is virtually indifferentiable in our case as the training takes very little time.

For DT, as the structure is very simple, average run time is very short (around some seconds). For NN, run time can vary depending on the dataset size, but on average it takes around ten seconds or so, with outliers (Pokerhand, Connect-4 and Covertype) taking around 40 seconds. For DNDT, in the same logic, only takes an average of around three seconds, with the same datasets taking much longer (up to three minutes and a half).

4 Results

Our results comparing accuracies is shown in the table below.

Overall, we see that this table indeed confirms the claim that DT gives the best overall performance. However, we cannot say that the experiment was completely successfully reproduced, because we see that, compared to the numbers in the paper, some of ours are off by a quite a few percent and consequently, some models were wrongfully declared better. We see this in, for example, Poker Hand, where in our experiment NN outforms better than DNDT whereas the paper observed a tie. However, we observed the same winning model for each of the dataset, considering accuracies within 1 percent of each other as a tie.

	DNDT	DT	NN
Iris	**100.0**	**100.0**	**100.0**
Haberman's Survival	72.6	66.2	**72.7**
Car Evaluation	85.0	**98.1**	91.4
Breast Cancer Wisconsin	92.6	89.1	**96.0**
Pima Indians Diabetes	71.4	**74.0**	70.8
Poker Hand	45.4	**81.4**	66.5
German Credit	75.5	66.8	**76.3**
Connect-4	66.5	**76.4**	73.5
Image Segmentation	73.2	**96.4**	92.2
Covertype	82.6	**93.6**	83.8

Table 2: Test set accuracy of each model

4.1 Results beyond original paper

4.1.1 Understanding the DNDT model

After conducting the experiments mentioned in the research paper we obtained results that reinforced the findings. The DNDT model, which differs from decision trees by using a binning function for making split decisions proved to be effective. This binning functions differentiability allows for backpropagation during training. By utilizing stochastic descent (SGD) for optimization we can simultaneously search for both the structure and parameters of the tree. Moreover, GPU acceleration greatly reduces training time especially when dealing with models. During the replicated experiments DNDT demonstrated its ability, to self prune at both split and feature levels indicating regularization. The model also displayed feature selection by disregarding important features. We applied this concept in our experiments. We observed significant improvements in accuracy across most datasets after removing certain features. To evaluate DNDTs performance we tested it on 10 datasets such as Iris, Habermans Survival, Car Evaluation and Breast Cancer Wisconsin. These datasets varied in size features included and classes represented. We compared DNDTs test accuracy with that of decision trees and neural networks, across these datasets.

For example, when it comes to the Iris dataset. DNDT achieved accuracy like the decision tree model. On the dataset Breast Cancer Wisconsin, DNDT recorded an accuracy of 92.6% slightly surpassing the decision trees 89.1%.We observed that as we increased the number of cut points (model complexity) DNDTs performance generally improved until it reached a plateau. This suggests that larger models do not tend to overfit. Additionally we noticed that the number of cut points decreased as their total number increased, which further indicates that the model has a self regulating nature. Regarding feature importance and selection we compared DNDTs approach with decision trees. In the case of the Iris dataset both models identified the feature as important. However on datasets, like Habermans we found differences in feature importance between DNDT and traditional decision trees.

To quantify how similar DNDTs and decision trees feature rankings are, we used Kendall's Tau metric. We found a moderate correlation overall. The replicated experiments confirm that not does DNDT outperform networks, in certain tabular datasets but it also competes effectively with traditional decision trees. Moreover it offers interpretability and ease of implementation. Some potential areas, for research include investigating how DNDTs ability to regulate itself works combining it with CNNs to enable learning and applying neural network based methods to facilitate the transfer learning of decision trees.

4.1.2 Hyperparameter Sensitivity Analysis

After conducting an analysis to understand how different factors affect the performance of the DNDT model, such as the number of cut points learning rate and binning function parameters, we found some results.

Firstly we noticed that increasing the number of cut points generally led to increased performance for the model. However there was a point where further increasing the cut points didn't improve its performance anymore. This suggests that there is a range for this parameter. On the hand we observed that changes in the learning rate and binning function parameters had impact on the models performance. It remained relatively stable across settings for these hyperparameters. Our findings indicate that as we increase complexity by adding points, the model can handle it without overfitting. This aligns with our observation that larger models don't tend to suffer from overfitting issues and have a design between complexity and generalization.

When it comes to learning rate and binning function parameters we found that they influence performance. The model performed consistently across datasets regardless of variations in these specific hyperparameters. This reinforces our belief that it is robustness. Overall our analysis highlights insights regarding how these factors affect the DNDT models performance while showcasing its ability to adapt to varying conditions. The user friendliness of the DNDT model is a characteristic as it reduces the requirement for hyperparameter tuning.

The outcomes of the hyperparameter sensitivity analysis align with our focus on simplicity in implementation. Due to its sensitivity to hyperparameters, the model simplifies the process of tuning making it more accessible for users who may not possess extensive expertise in hyperparameter optimization. This quality, combined with the models interpretability, presents DNDT as a choice for applications particularly, in domains where transparency and ease of use are essential.

5 Discussion

Experimental Results and Paper Claims: Our experiments replicated the claims of the paper to a certain extent. We observed that DT performed better than DNDT and NN across the 10 datasets. DT had the highest accuracy for seven of the datasets, followed by NN with four wins followed by DNDT with three wins (considering Haberman and German Credit datasets as ties). Among of the 10 datasets, our experiment for the German Credit dataset had NN with the highest accuracy with 76.3%, and the next highest DNDT with 75.5%; the paper found that NN and DNDT are tied with 70.5%. Although our experiments do not replicate the identical acccuracies, the accuracies are 0.8% apart; we consider this a tie. Similarly, Haverman's Survival is tied in the paper with 70.9%, in our difference in the two highest is 0.1%, we also consider this a tie.

Unfortunately the claim that DNDT is better than NN is difficult to validate using the 10 datasets. By only looking at our experiments, NN performs better than DNDT. The calculated mean reciprocal ranks are 0.81, 0.64 and 0.53 for DT, NN and DNDT respectively. However, the datasets that we did not implement, specifically Titanic, Flight Delay and HR Evaluation contained two additional wins for DNDT and one for DT. As such, considering this, DNDT would surpase NN in the number of wins. As a result, we are unable to confirm the paper's claim that DNDT is better than NN.

Additionally, the margin of superiority is not the same. There is high variability for the Car Evaluation, Poker Hand, and Image Segmentation datasets. The highest different is for Image Segmentation, our implementation for NN scored 2^{nd} place with accuracy of 92.2% compared to the paper where NN scored 3^{rd} place with accuracy of 47.5%. This is a difference of 44.7%, which is significant, and is likely due to implementation details or potential data set differences. The overall result is that the effectiveness of the models is context-dependent, and that different variants of a dataset can have different effects on each model.

Strengths and Weaknesses of Our Approach: The main strength of our approach is replicating most of the datasets. This allowed for a robust test into each of the model's performance and insight into the paper's claims. We originally planned to replicate all the datasets but were unable to find the same datasets with exactly the same characteristics. For example, there are many Titanic datasets with varying feature dimensions; we were unable to find the paper's one. Our experiments strengthened some of the paper's claims, specifically that DT is better than NN and DNDT, but also highlights further areas of research. Some more investigation should be performed for the reason behind the accuracy inconsistencies between our implementation and the paper's. Additionally, although we mentioned it a strength, we implemented many datasets but implementing more would be beneficial. Unfortunately, we were unable to find similar

datasets to the papers and are limited to the paper's datasets only (since this is a reproduction study). As such, this is the original paper's weakness and it is a major one. Ideally, enough datasets are used until each model's behaviour is clearly understood.

Implications and Future Directions: Our attempt at replicating the paper's results emphasizes the importance of detailed reporting in machine learning, especially in terms of model parameters and dataset sources. The variation in our results compared to the paper most likely would not exist if the original paper had rigorous documentation regarding all the models and datasets. Furthermore, an interesting future area of research is further investigating the balance between interoperability and performance of DNDT. Potential applications/industries would be interesting to explore. Additionally, further tuning the parameters of the DNDT model to achieve higher accuracy in the datasets where is scored low would be beneficial to further understanding the models effectiveness and applicability.

5.1 What was easy

The model easiest to implement was the DT as it only needs one line of code to get a model object. NN was also easy to implement, since setting up the network takes around seven lines of code. DNDT was also easy implement, and did not require special attention. Although some processing needed to be done for the datasets, implementing all the models was straightforward.

5.2 What was difficult

Firstly, one might notice that we have discarded some datasets; this is because they were either not available anymore, or had too many under the same name. Secondly, as mentioned previously, a lot of details concerning the structure of the DNDT and NN models were left unknown to the reader, making it difficult to try and match the experiments' accuracies. In the end, we had to resort to semi-blind guesses to see if our results got more precise or not. While these troubles were not overwhelmingly challenging, it still gave us some troubles. Lastly, what we truly felt was confusing was the official given code. The model code was understandable, but the example demo had little to no comments, making us question what was going on.

Another difficulty, which is also a frustration, is the lack of documentation of the paper in terms of the hyper-parameters for the model. We were unable to replicate all the accuracies due to unknown parameters and missing random states for the data splits. We noticed that from run to run, accuracies would vary up to 10% due to the data split for test and validation data being random. If the authors provided the random state for each of the datasets, we believe that replicating the exact accuracies would be possible. This is important, because it validates the analysis of the authors but also allows for expansion of the original claims. Expanding a claim which cannot be entirely replicated is difficult and leads it to have questionable validity.

5.3 Communication with original authors

We did not have any contact with the original authors as it did not occur to us that we could, especially on short notice.

5.4 Statement of contribution

Joyce Liu - DT implementation and report

Yoah Qiu - DNDT implementation and report

Mihail Calitoiu - NN implementation and report

References

1. Yang, Y., Morillo, I. G.; Hospedales, T. M. (2018, June 19). Deep Neural Decision Trees. arXiv.org. https://arxiv.org/abs/1806.06988