

Plotly.plus, an Improved Dataset for Visualization Recommendation

Group 24 – EDDS – Experiment reproducibility

Jonathan Cazco Gonzalez

Data Science, TU Wien, Wien, Austria, e12227662@student.tuwien.ac.at

Ivana Dasovic

Data Science, TU Wien, Wien, Austria, e12230009@student.tuwien.ac.at

Stefania Dimancea

Data Science, TU Wien, Wien, Austria, e12229964@student.tuwien.ac.at

Andrei Tosa

Data Science, TU Wien, Wien, Austria, e12228508@student.tuwien.ac.at

ABSTRACT

In this paper, we aim to examine two different machine learning modes based on convolutional neural network algorithms, trained to distinguish and classify insightful visualizations. The technical mechanism specialized in filtering visual data is based on a Visual Recommendation System (VRS), constructed to evaluate data features and provide image assessment for labeling visualizations. The objective of original the experiment is to augment the original Plotly dataset by applying plot filtering optimization to construct a new corpus of images collected within an improved version, defined as Plotly.plus dataset. For conducting the experiment, we have implemented Chart Classification using a simplified VGG model and a Multi-level classification using convolutional neural networks.

CCS CONCEPTS

• Information systems → Clustering; Data cleaning; • Human-centered computing → Visual analytics; Information visualization; • Computing methodologies → Cluster analysis; Neural networks.

KEYWORDS

Visualization Recommendation; Data Analytics; Plotly; Image classification; Multi-label classification; Machine learning for Visual Analytics (ML4Viz); Convolutional neural networks; Deep learning

1 Introduction

Information visualization covers valuable techniques used to illustrate insights for a broad range of data analysis subjects. The process gathers interactive elements to create a powerful tool to display the meaning behind the data. The action of classifying visualizations is conducted under the strategy of training machine learning algorithms to classify various types of visualizations. In one part, the experiment is focused to learn the visualizations directly from the data, recommending the most suitable visualization that can be generated from a given data collection. As with any machine learning system, the models require a coherent and possibly large dataset of insightful visualizations to train the neural network. The initial Plotly dataset is lacking several important key points as data is biased due to a lack of expert consistency of adequate examples. The solution proposed was to create a new dataset, Plotly.plus that contains a corpus of visualizations defined by specific and accurate features as the result of training the neural network on different environmental parameters and improves the image processing models' classification rate.

2 Methods

Deep learning methods are a component of machine learning techniques that use artificial neural networks with multiple layers to model complex relationships between inputs and outputs. Source code developments are not provided in the official research papers. Following the referenced papers' instruction, we reconstructed from scratch two convolutional neural network (CNN) methods as a deep learning technique to study the classification of graphic visualizations. Specific to the dataset used, images often have high dimensional size, making it computationally inefficient to use raw image data as inputs for a neural network. The convolutional algorithm mitigates this problem by extracting the most significant features from an image and using only those features as input.

2.1 Model replication

The models used for these experiments were re-constructed following the respective architecture descriptions in [1] and [2], but also including the modifications made by the authors of the target paper. The first model's name is "model_81." This model is a simplified version of the VGG16 CNN with only 5 layers, thus renamed VGG5 by the authors. The Keras implementation was chosen for the replication. As additional changes, the input layer is set to accept an input shape of (96×96) images in 3 color channels, while the original multiclass output layer is replaced by a SoftMax activation function with 2 output channels for the binary classification. Thus, also selecting the binary cross entropy as loss function when the model is compiled.

The second model reproduced, noted as "model_97." is a multi-level classification algorithm, implemented using PyTorch. The algorithm is divided into the following parts: one iteration is executed for Label Powerset and the second iteration is executed for Binary Relevance. Each image is iterated by applying Canny edge detection to increase the classification performance of the method. For training the CNN model, the images were resized to a fixed dimension of 120×120 pixels. The neural network layer is activated with the rectified linear function (ReLU), used to introduce non-linearity into the network, allowing it to model complex relationships between inputs and outputs. The fundamental parameter characteristics remained the same as in the previous model, by operating Adam optimizer and a cross-entropy loss function. The models architecture can be summarized in the following tables:

Layer (type)	Output shape	Number of parameters
Conv2D	None, 96, 96, 32	896
Activation	None, 96, 96, 32	0
BatchNormalization	None, 96, 96, 32	128
MaxPooling2D	None, 32, 32, 32	0
Dropout	None, 32, 32, 32	0
Conv2D	None, 32, 32, 64	18496
Activation	None, 32, 32, 64	0
BatchNormalization	None, 32, 32, 64	256
Conv2D	None, 32, 32, 64	36928
Activation	None, 32, 32, 64	0
BatchNormalization	None, 32, 32, 64	256
MaxPooling2D	None, 16, 16, 64	0
Dropout	None, 16, 16, 64	0
Conv2D	None, 16, 16, 128	73856
Activation	None, 16, 16, 128	0
BatchNormalization	None, 16, 16, 128	512
Conv2D	None, 16, 16, 128	147584
Activation	None, 16, 16, 128	0
BatchNormalization	None, 16, 16, 128	512
MaxPooling2D	None, 8, 8, 128	0
Dropout	None, 8, 8, 128	0
Flatten	None, 8192	0
Dense	None, 1024	8389632
Activation	None, 1024	0
BatchNormalization	None, 1024	4096
Dropout	None, 1024	0
Dense	None, 2	11275
Activation	None, 2	0

Layer type	Output shape	Trainable parameters
Conv2D	(120, 120, 32)	320
Conv2D	(118, 118, 32)	9248
MaxPooling2D	(59, 59, 32)	0
Dropout (25%)	(59, 59, 32)	0
Conv2D	(59, 59, 64)	18,496
Conv2D	(57, 57, 64)	36,928
MaxPooling2D	(28, 28, 64)	0
Dropout (25%)	(28, 28, 64)	0
Conv2D	(28, 28, 64)	36,928
Conv2D	(26, 26, 64)	36,928
MaxPooling2D	(13, 13, 64)	0
Dropout (25%)	(13, 13, 64)	0
Flatten	(10,816)	0
Dense	(512)	5,538,304
Dropout (50%)	(512)	0
Dense (Softmax)	(4)	2052

Figure 1: Architecture summary for models 81 (left) and 97 (right)

2.2 Reproducibility challenges

The first challenge encountered was the non-existing code repository that accompanies the selected paper as a basis for the replication setup. Thus, the code had to be started from scratch, including the processing of the images and the work pipeline setup. A subsequent challenge deals with the issue that even though the models' architectures are well described in the referenced paper, there is no information on the (optimal) parameters used to tune/compile the CNNs. For both implementations, the standard parameters from the derived algorithms or the default values of the library used were considered. Finally, the last challenge was related with the computer resources available to reproduce the experiment, resulting in prolonged wait times to achieve partial results.

3 Experiment reproduction results

For the first model, *model_81*, the implementation paper stated that the computing power adequate for training the network was similar to a normal laptop (core i7, 8Gb RAM). However, the target paper did not report the computer specifications, but stated that training should be completed within an hour on a "standard" computer. Despite these claims, runtime was an issue, as it took approximately 20 minutes plus the training time over technical setup for 10 epochs which leads to more than 3 hours of execution when 10- fold cross validation is included. The number of epochs was also not reported. Unfortunately, we were unable to fully reproduce the results of the original study as the building of the second model, *model_97*, proved unsuccessful. The lack of code and quality documentation hindered our ability to construct the model, resulting in an inability to reproduce any results associated with it.

4 Evaluation

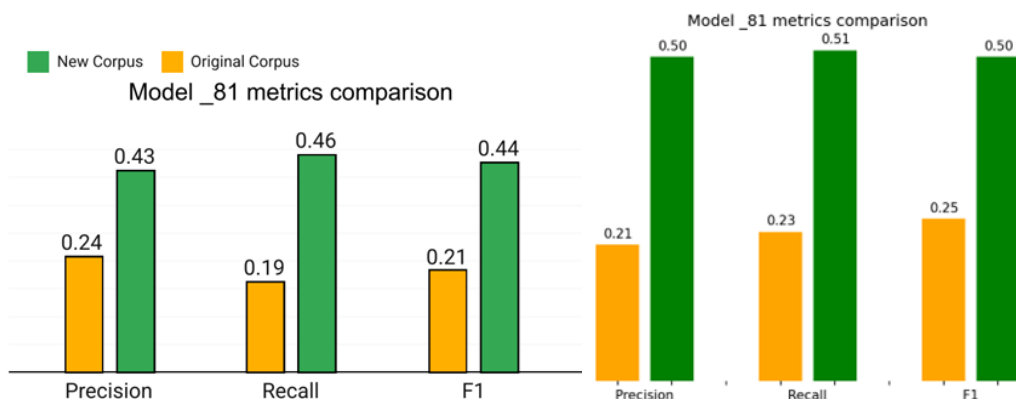


Figure 2: Results from the original study.

Figure 3: Reproduced results.

The original Plotly dataset was tested against the Plotly.plus dataset. The evaluation of the classification results was performed using precision, recall and F1 score. Figure 2 demonstrates results from the paper and Figure 3 represents our replicated results. Yellow bars indicate results obtained using the original corpus (Plotly dataset), while green bars show results obtained by models trained on a new corpus (Plotly.plus dataset). Our results show a strong resemblance to those presented in the paper, despite a slight difference due to the unavailability of the tuning parameters, such as the number of epochs, used in the original study. The hypothesis that the new corpus leads to improved model performance was also confirmed through our results.

5 Conclusion

Reproducibility is a key aspect of scientific research, as it allows other researchers to validate the results of a study and build on existing work. When results are reproducible, it increases trust in validity of the findings and confirms that the methods and techniques used were robust and reliable. Additionally, reproducing the results of a project can lead to new insights, discoveries, and advancement in the field of study. By supporting replication studies and increasing the credibility of research results, reproducibility plays an essential role in advancing scientific knowledge and promoting scientific progress.

The targeted study could be considered as replicable, yet having into account the challenges presented, it's difficult to get exactly the same results as the ones provided in the paper. The reason for that lies in the fact that the code used in the research isn't provided. We successfully partially replicated the results from the original study. Our results obtained using model_81 were consistent with those reported in the paper. Unfortunately, we were unable to reconstruct model_97, and as a result, we couldn't reproduce the results obtained from this model.

In the context of reproducibility, parameters play a crucial role in achieving consistent and accurate results as they greatly influence the final outcome. Beside model description and identification of libraries used, there is no additional information on tuning parameters which can substantially impact the performance of models. Significantly longer execution than described in the paper could indicate that models are either badly documented and some of the information aren't correct or that we interpreted the description in a wrong way and built a model that differs from the original one. On the other hand, the data used in the original experiment was openly available and easily accessible and the challenge of data unavailability and inaccessibility was avoided. Consequently, we successfully replicated results for model_81, but were unable to reproduce results for model_97 due to the absence of proper code, documentation and computer power.

REFERENCES

- [1] Luca Podo, Paola Velardi. 2022. Plotly.plus, an Improved Dataset for Visual Recommendation. 31st ACM International Conference on Information & Knowledge Management, 1 (October. 2022), 36-44. DOI: <https://doi.org/10.1145/3511808.3557669>
- [2] Filip Bajić, Josip Job, and Kresimir Nenadić. 2019. Chart classification using simplified VGG model. In 2019 International Conference on Systems, Signals and Image Processing (IWSSIP). IEEE, 229--233.
- [3] Cem Kosemen and Derya Birant. 2020. Multi-label classification of line chart images using convolutional neural networks. SN Applied Sciences 2, 7 (2020), 1–20.