Automated image classification with expandable models

Andre Henriques, 6644818

Contents

1 Introduction

Currently, there are many classification tasks that are being done manually. These tasks could be done more effectively if there was tooling that would allow the easy creation of classification models, without the knowledge of data analysis and machine learning models creation. The aim of this project is to create a classification service that has 0 requires zero user knowledge about machine learning, image classification or data analysis. The system should allow the user to create a reasonable accurate model that can satisfy the users' need. The system should also allow the user to create expandable models; models where classes can be added after the model has been created.

1.1 Aims

The project aims to create a platform where users can create different types of classification models without the users having any knowledge of image classification.

1.2 Objectives

This project's primary objectives are to:

- Create platform where the users can create and manage their models.
- Create a system to automatically create and train.
- Create a system to automatically create and train models.
- Create a system to automatically expand and reduce models without fully retraining the models.
- Create an API so that users can interact programatically with the system.

This project extended objectives are to:

- Create a system to automatically to merge modules to increase efficiency
- Create a system to distribute the load of training the model's among multiple services.

2 Literature and Techincal Review

2.1 Alternatives to my Project

There currently exist systems that do image classification, like Google Vision AI[1], and Amazon's Rekoginition[2]. Their tools, while providing similar services to what my project is supposed to do, it mostly focusses on general image classification rather than specific image classification, i.e. Car vs Boat, vs, Car model X vs Car model Y.

2.2 Creation Models

The models that I will be creating will be Convolutional Neural Network(CNN)[3], [4]. The system will be creating two types of models that cannot be expanded and models that can be expanded. For the models that can be expanded, see the section about expandable models. The models that cannot be expanded will use a simple convolution blocks, with a similar structure as the AlexNet[5] ones, as the basis for the model. The size of the model will be controlled by the size of the input image, where bigger images will generate more deep and complex models. The models will be created using TensorFlow[6] and Keras[7]. These theologies are chosen since they are both robust and used in industry.

2.3 Expandable Models

The current most used approach for expanding a CNN model is to retrain the model. This is done by, recreating an entire new model that does the new task, using the older model as a base for the new model[2], or using a pretrained model as a base and training the last few layers.

There are also unsupervised learning methods that do not have a fixed number of classes. While this method would work as an expandable model method, it would not work for the purpose of this project. This project requires that the model has a specific set of labels which does not work with unsupervised learning which has unlabelled data. Some technics that are used for unsupervised learning might be useful in the process of creating expandable models.

3 Technical overview

3.1 Web Interface

The user will interact with the platform form via a web portal. The web platform will be designed using HTML and a JavaScript library called HTMX[8] for the reactivity that the pagers requires. The web server that will act as controller will be implemented using go[9], due to its ease of use. Go was chosen has the programming language used in the server due to its performance, i.e. [10], and ease of implementation. As compiled language go, outperforms other server technologies such as Node.js. Go also has easy support for C ABI, which might be needed if there is a need to interact with other tools that are implemented using C. The web server will also interact with python to create models. Then to run the models, it will use the libraries that are available to run TensorFlow[6] models for that in go.

3.2 Creating Models

The models will be created using TensorFlow[6]. TensorFlow was chosen because, when using frameworks like Keras[7], it allows the easy development of machine learning models with little code. While tools like PyTorch might provide more advanced control options for the model, like dynamic graphs, it comes at the cost of more complex python code. Since that code is generated by the go code, the more python that needs to be written, the more complex the overall program gets, which is not desirable. The original plan was to use go and TensorFlow, but the go library was lacking that ability. Therefore, I chose to use python to create the models. The go server starts a new process, running python, that creates and trains the TensorFlow model. Once the training is done, the model is saved to disk which then can be loaded by the go TensorFlow library.

3.3 Expandable Models

The approach would be based on multiple models. The first model is a large model that will work as a feature traction model, the results of this model are then given to other smaller models. These model's purpose is to classify the results of the feature extraction model into classes. The first model would either be an already existent pretrained model or a model that is automatically created by the platform. The smaller models would all be all generated by the platform, this model's purpose would be actually classification. This approach would offer a lot of expandability, as it makes the addition of a new class as easy as creating a new small model.

4 Workplan

4.1 Timeline

Month	Goals
September	• Experimenting with web development frameworks. • Started working on code development.
October	• Starting working on Project Synopsis. • Continue working on project development. • Finish user management system and basic ui decisions. • Finish data upload section of the website.
November	• Finish writing on Project Synopsis. • Finish coding the basic model generation and training.
December	• Improve basic model generation.
January	• Add api support. • Started working on the final report
Feburary	• Start working on expandable models generation
March	• Create systems to expand the expandable models and contract models • Review draft submissions
April	• Basic final report finish • Create systems to expand and reduce expandable models
May	• Finish and submit final report

4.2 Risks

Risk	Mitigation
Automatic model generation is not feasable	Other methods for creation of model can be used i.e. transfer learning, or semi-unsupervised learning
Easy model expancion is not feasble	Other methods of updating the models need to be implemented i.e. retratinng the entire model.
Not enough compute power to train models fast enough to develop the program	Free credits from online cloud providers like google or amazon

A References

[1] Google. "Vision ai — google cloud." (2023), [Online]. Available: https://cloud.google.com/vision?hl=en.

[2] Amazon, "Image recognition software - ml image & video analysis - amazon rekognition - aws," 2023. [Online]. Available: https://aws.amazon.com/rekognition/.

[3] Y. LeCun, B. Boser, J. Denker, *et al.*, "Handwritten digit recognition with a back-propagation network," *Advances in neural information processing systems*, vol. 2, 1989.

[4] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[6] Martín Abadi, Ashish Agarwal, Paul Barham, *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015. [Online]. Available: https://www.tensorflow.org/.

[7] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.

[8] *</> htmx - high power tools for html*, [Online; accessed 1. Nov. 2023], Nov. 2023. [Online]. Available: https://htmx.org.

[9] *The Go Programming Language*, [Online; accessed 1. Nov. 2023], Nov. 2023. [Online]. Available: https://go.dev.

[10] *A journey from Node to GoLang*, [Online; accessed 5. Nov. 2023], Nov. 2023. [Online]. Available: https://www.loginradius.com/blog/engineering/a-journey-from-node-to-golang.