CS230

Electrical Wire Connector Classification using Transfer Learning

Boyeon Kim[*]
Department of Mechanical Engineering
Stanford University
`bokim@stanford.edu`

1 Introduction

1.1 Motivation

Electrical wire connectors are widely used in daily life, though it might not seem obvious to the users. All electronic devices, such as computers, cars, cellular phones and even electronic kettles, are built with numerous electrical wires and connectors. For the purpose of connecting and disconnecting electrical components and PCBs from each other in the process of assembling the devices, troubleshooting the hardware malfunctioning problems, and replacing the broken parts, electrical components are usually connected with electrical connectors, rather than soldered on as a one body.

Figure 1: Various types of electrical wire connectors

The shape of an electrical wire connector can be quite unique. However, there is no written or engraved parts number or the manufacturer's name on it, most likely due to its small size. Accordingly, it is very hard to look for its information or replacement unless one contacts the device's manufacturer. Moreover, very often the manufacturer does not even want to disclose the detailed hardware information to prevent users from damaging the device while trying to modify or fix it or to make the users purchase the replacement parts from itself.

Therefore, for most of users who are generally lack of hardware knowledge, it will be hard to look for the right connectors without knowing the names. This will cause a lot of searching hours on google images. Especially for super-users or engineers who would like to tweak the devices' original usage, repair their own devices by themselves, or build their own devices with existing electrical modules, looking for the correct connectors is crucial, but tricky and time-consuming. Even for the other users who are lack of general knowledge of power connectors

To facilitate the connector-searching process, I would like to train a deep learning model

[*]Bo is a PhD candidate in Mechanical engineering department. The main research interest is in robotics hardware and multi-linked robot control.

with various electrical wire connector RGB images and build a classifier that predicts corresponding the types of the connectors.

2 Related work

Literature search shows that CNNs has been widely used to classify small or microscopic objects by extracting their distinctive features, including, but not limited to small and indistinguishable hardware components[1], [2], [3]. Even with small size data, similar objects can be classified using transfer learning and extra added network [4], [5]. Especially image augmentation along with transfer learning can dramatically increase the prediction [5], [6],[7]. Also shown that ResNet-50 among VGG-16, VGG-19, and ResNet-50 produced the best performance overall [8].

3 Dataset and Features

3.1 Data Collection

Unfortunately, there is no existing dataset available for the classification work. Due to time constraint and the massive number of connector types, it was impossible to generate a dataset of the connector images from various companies. Therefore, I decided to focus on classifying Wire-to-Wire Connectors of Molex LLC only, which has a large market share. Also given that the Molex connectors in different types are very similar to each other, if the classification of the Molex connectors is feasible, the classification of the other connectors should be also feasible.

Figure 2: Several classes of Wire-to-Wire connectors of Molex LLC (Ditto, Micro TPA, Milli-Grid, Nano-fit, PicoBlade, S062, and SL from left)

The input images along with the part numbers and the type of connectors were collected from the official Molex website. Due to lack of image data, only the following classes of connectors were used in the test: PicoBlade Connector System, Milli-Grid Connector System, MicroTPA Connector System, SL Modular Connectors, Ditto Wire-to-Wire Interconnects, and Nano-Fit Power Connectors. Although the original idea was to classify each of a connector's name, due to lack of data and for the purpose of showing the algorithm's feasibility, I decided to classify the connectors in their classes.

Class:	Milli-Grid	MicroTPA	SL	S062	Ditto	Nano-Fit	PicoBlade
Image count:	5	4	27	60	21	207	42

Table 1: Number of image data per each class

3.2 Data Preprocessing

3.2.1 Augmentation

There are only one image per each connector in isometric view is available in the official Molex website. To overcome the barriers to learning on this small size of the dataset, data augmentation was implemented. Considering most of the connectors has a symmetric shape, flipping images horizontally, rotating, and zooming augmentation was applied to the train dataset.

Figure 3: Image augmented result

3.3 Normalization

For secure faster training time, each images were reshaped into 64 by 64 pixels. And then, each image pixel values were re-scaled and normalized from 0-255 range to 0-1 range.

3.4 Data Split

Since the data size is very small, I set train-dev-test ratio as 80-10-10.

4 Methods

The transfer learning on a pretrained CNN with added extra fully connected networks chain blocks and softmax activation at the end was chosen to be the main algorithm of the training due to small data size. Specifically ResNet50 architecture has been chosen since ResNet is specifically built for image recognition and there are numerous papers shown that it is the latest successful and outperforming approach. ResNet with its residual block and skip connection usage prevents over-fitting or vanishing gradient for deeper neural network.

In training, the baseline of this experiment's architecture is pretrained ResNet50 on ImageNet. Since the pretrained network had learned various visual features and especially the earlier layers consists more basic features, such as shapes and edges, it is feasible to fine-tune and train the network with very small data size. To check how much features the algorithm can effectively get from the pretrained ResNet50, the various freezing weights range were tested.

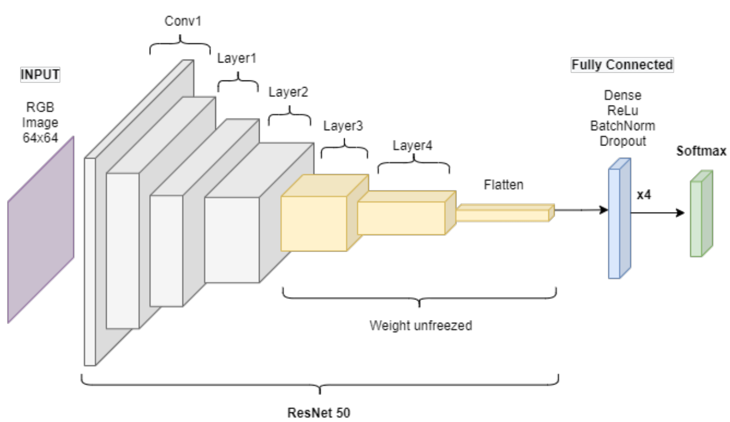

Figure 4: Model architecture: Transfer learning on pretrained ResNet50

As shown in figure 4, I added 4 extra fully connected layer blocks, which consist dense, relu activation, batch-normalization, and dropout in the order, and the softmax layer at the end. Also the hyperparameters-learning rate, batch normalization momentum, and dropout rate-were adjusted for the better performance. Tuning the gradient descent learning rate was done first to fit the rate onto the graphical shape of the network structure and to find the best. After that, the others were adjusted.

For the optimizer, I chose Adam optimizer which has advantages of both MomentumProp, and RMSprop. Therefore, the model can have stable learning rate for each parameter and be affected less by any noise. The loss was simply calculated with cross-entropy, a commonly used loss function. Validation was performed after every epoch to evaluate the learning while training the network. Also,, To improve accuracy, I will perform manual error analysis to check the variance and bias and see which part of dataset are more vulnerable. The implementation was done with Keras.

$$\mathcal{L}(y^{(i)}, \hat{y}^{(i)}) = -\sum_{j}^{C} y^{(i)} \log(\hat{y}^{(i)}), L = \sum_{i=1}^{m} \mathcal{L}(y^{(i)}, \hat{y}^{(i)})$$

5 Experiments/Results/Discussion

5.1 Resnet50 Weight Freezing Range

First I started training with the Molex dataset with ResNet50 with added extra fully connected layers and adjusted softmax, label size 7, with all ResNet50 pretrained weights fixed. The training accuracy was increasing nearly up to 90 percent, however the validation accuracy had a tendency to stop around 70 percent, with different hyperparameter adjustments. The problem was due to freezing the entire weights of ResNet50. As the images of the Molex connectors are constructed with simple lines, shapes and color, the later layers fixed weights were preventing the model to have a good prediction. This also lead the data to overfit and produced the low validation prediction rate. After testing several times with different ranges of weight freezing range, it was shown that freezing only layer 1 and 2 shown in figure 4, or up to layers 80 in Keras, was optimal for the Molex dataset.

5.2 Hyperparameters

After fixing the weight freezing, I tuned the model over three parameters: the learning rate, the learning decay rate, dropout rate, and batch normalization momentum. I tuned the parameters by random search, sampling the learning rate over [1-e0, 1-e7], learning rate decay over [0.7, 1], dropout rate [0.4, 0.7], and momentum [0.9, 0.99]. After evaluating the result, I reduced the search range to find the optimum. The final picked hyperparameter is the learning rate 3e-4, decay rate 0.9, dropout rate 0.5, and momentum 0.95, as shown in figure 5. The model with other similar parameters were also able to achieve similar results.

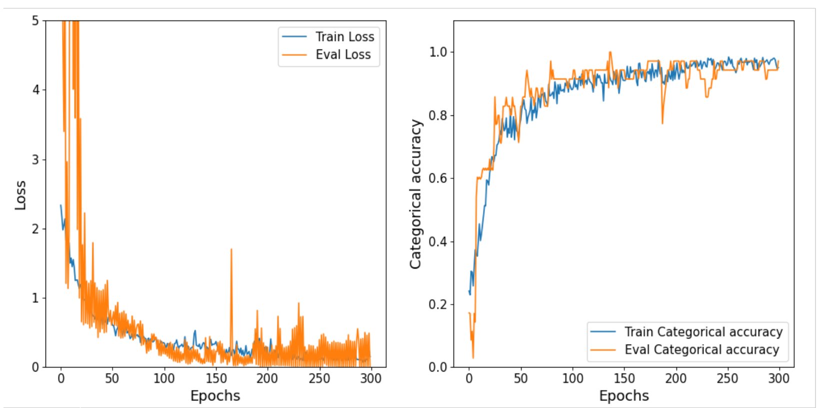

Figure 5: Final model train and validation result

Train accuracy	Val accuracy	Test accuracy
0.9741	1.0	1.0

Table 2: Final prediction accuary result

5.3 Error Analysis

The confusion matrix on the left in 6 shows that most of data mismatching was caused by small image data size of Milli-Grid and MicroTPA classes. By removing the two data piles from the dataset, the performance of the model has significantly increased. This shows that even with augmented data implementation increasing data size is crucial to improve the model.

6 Conclusion and Future Work

The biggest barrier of this approach is the size of the dataset. However, with the image augmentation and transfer learning, the model was still able to classify the classes of the connectors. Interestingly, although the Molex connectors' shapes and colors are very similar to each other and in the image it is

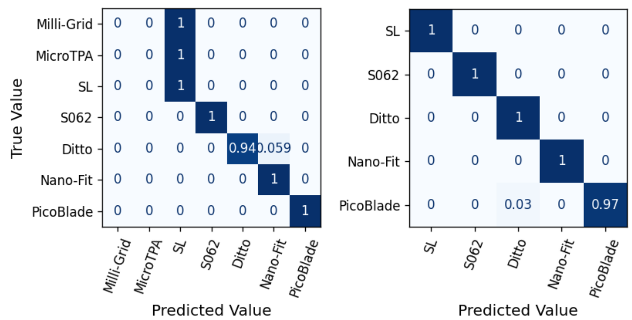

Figure 6: Left: with 7 classes, Right: with 5 classes

hard to know the actual size of the connector, with the pretrained large CNN network and its weights and adjusted parameters, feature detection worked well to achieve 100% test prediction rate. Given it is very hard to identify the connector types just by observing them on human-level, this performance is outstanding. This shows that this approach is very promising to detect other more diverse shapes of connectors. However, given that validation and testing were done only with the official Model website images, it is possible that the model might not work well with the actual users' images.

The original idea was to identify not only the class of the connectors, but also the connectors' model numbers. For that ultimate goal, there are several options to make this model more robust. First, increasing the data size would help with better classification process. Second, with the larger dataset, build a multi-input model. This has been already tried with my dataset. However, since the classification was done on "types" of connectors, not on "model numbers", inputting other information-such as plug, receptacles, vertical headers, horizontal headers, surface headers, number of rows, pin materials, and temperature resistiveness-was not effective, rather decreased the accuracy. Accordingly, if the model is to classify model numbers, multi-input model is very promising. Third, build multiple CNNs and merge at the end to identify different features of the connectors. The colors of connectors usually match with its material properties. And the pin hole numbers is a most significant feature to detect. These properties are important to identity the "type" and its "model number". Therefore, if there is a model has multiple different CNNs to detect these different features separately and then merge the output at the end, it would be possible to achieve the goal.

Code: `https://github.com/bonauzer/cs230_project`

References

[1] C. Pramerdorfer and M. Kampel, "A dataset for computer-vision-based PCB analysis," 2015 14th IAPR International Conference on Machine Vision Applications (MVA), 2015, pp. 378-381, doi: 10.1109/MVA.2015.7153209.

[2] Lu, Hangwei Mehta, Dhwani Paradis, Olivia Asadizanjani, Navid Tehranipoor, Mark Woodard, Damon. (2020). FICS-PCB: A Multi-Modal Image Dataset for Automated Printed Circuit Board Visual Inspection.

[3] Reza, M.A., Chen, Z. Crandall, D.J. Deep Neural Network–Based Detection and Verification of Microelectronic Images. J Hardw Syst Secur 4, 44–54 (2020). https://doi-org.stanford.idm.oclc.org/10.1007/s41635-019-00088-4

[4] P. Borwarnginn, K. Thongkanchorn, S. Kanchanapreechakorn and W. Kusakunniran, "Breakthrough Conventional Based Approach for Dog Breed Classification Using CNN with Transfer Learning," 2019 11th International Conference on Information Technology and Electrical Engineering (ICITEE), 2019, pp. 1-5, doi: 10.1109/ICITEED.2019.8929955.

[5] Barman, Rahul, et al. "Transfer learning for small dataset." Proceedings of the National Conference on Machine Learning, Mumbai, India. Vol. 26. 2019.

[6] T. Zhang and X. Zhang, "Shipdenet-20: An only 20 convolution layers and <1-mb lightweight sar ship detector," IEEE Geoscience and Remote Sensing Letters, pp. 1–5, 2020.

[7] M. Kisantal, Z. Wojna, J. Murawski, J. Naruniec, and K. Cho, "Augmentation for small object detection," CoRR, vol. abs/1902.07296, 2019.

[8] Yuqing Gao and Khalid M. Mosalam. Peer hub imagenet: A large-scale multiattribute benchmark data set of structural images. Journal of Structural Engineering, 146(10):04020198, 2020.

[9] Francois Chollet et al. Keras, 2015.