CS230

CS230 Final Project: Automatic Laundry Sorting

Andy Liang, Hiroto Yamamoto
Stanford University
andy223@stanford.edu, hiroyams@stanford.edu

Abstract

Folding laundry has remained roughly the same for the past few decades, being a tedious and labor-intensive task. This project aims to develop an algorithm for distinguishing between 14 different types of clothing given a database of 4961 images. We explored three different methodologies to conduct this classification: a regular convolutional neural network approach, a transfer learning approach with ResNet50, and a supervised contrastive learning approach. The supervised contrastive learning method ended up being the most effective with a training accuracy of 81% and a test accuracy of 63%, while the other two methods trailed behind. Due to the small sample size and insufficient standardization of the dataset, our models were subject to overfitting.

1 Introduction

As college students living in dorms, we face the problem of dealing with chores while surviving through the academic challenges in school. This motivated us to think of ways to automate our weekly chores, specifically laundry.

Laundry has increasingly become more convenient, due to commercial washing and drying machines. The global market for Commercial Laundry Machinery was estimated at $4.4 Billion in the year 2022 and is projected to reach a size of $5.9 Billion by 2026 (Global Industry Analysts). However, folding laundry has remained roughly the same for the past few decades, being a tedious and labor-intensive task. We propose the development of an automated machine, where users can fold large amounts of laundry quickly and easily. This can be particularly useful for services like hotels that deal with large amounts of laundry, and people with busy schedules, disabilities, or limited mobility.

An automated folding machine would require specialized hardware with sensors, cameras, and mechanical arms, with a robust multi-class image classification algorithm to accurately identify and manipulate items of laundry. This project aims to develop such an algorithm, for distinguishing between different articles of clothing given an image. The input to the algorithm is images of clothing. We then use a neural network to output the category of the clothing, allowing us to sort pieces of laundry from images.

A major challenge to the proper classification of multiple clothing types is that different articles of clothing may look significantly different, but be of the same class. For instance, two shirts could have significantly different designs, colors, and sizes. This is in contrast to many other image classification problems, like handwriting identification with the MNIST dataset, where two images of the same class will be quite similar, even being written slightly differently. This presents a challenge to accurate classification, which we will discuss in this study.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

There are several methods for performing a multi-class image classification problem. CNNs have been shown to be effective at learning complex and nuanced visual patterns from the data, allowing them to accurately classify images into multiple classes. This has been demonstrated in various studies, which have applied CNNs to different datasets and tasks. For example, Krizhevsky et al. used a CNN to achieve state-of-the-art performance on the ImageNet dataset, which contains millions of labeled images from 1000 different classes. Another early application of successful CNN was to classify handwritten digits called the MNIST (Modified National Institute of Standards and Technology) database. With 70,000 monochrome handwritten digits that have been size-normalized and centered in a fixed-size image, CNN showed great accuracy for classification. Ezat et al. compared the performance of CNN models to other classification algorithms, showing that CNN performed better than the super-vector coding and support vector machines (SVM).

Another method for multi-class image classification is to use transfer learning. This involves the use of a pre-trained model on an enormous dataset and then fine-tuning it on a smaller dataset for the specific task at hand. He et al. proposed the ResNet model, which introduced the concept of skip connection and achieved improved performance on a variety of image classification tasks.

Reddy and Juliet used ResNet to classify Malaria cell images. Using 27,558 images of infected and uninfected cells, the input images were given to the ResNet50, with a batch size of 100. They used the Stochastic Gradient Decent (SGD) for their optimizer and Cetegorial-Cross Entropy for their loss function, giving an accuracy of 95.4% (Reddy and Juliet).

There has been various studies using different pre-trained models in transfer learning. Lu et al. used AlexNet to classify brain images as normal or abnormal. Jain et al. proposed a deep transfer learning-based Alzheimer's disease diagnosis system, using the pre-trained Vgg-16 model. Talo et al. implemented a CNN for a 5-class brain disease detection using MRI images. This study compared performances using AlexNet, Vgg-16, ResNet-18, ResNet-34, and ResNet-50. The study showed that ResNet50 obtained the best classification accuracy of 93.23%, while the lowest classification performance was the AlexNet model with 80.11% accuracy (Talo et al.).

Supervised Contrastive learning has also been applied to multi-class image classification with promising results. Khosla et al. described a method in which they extended the traditional unsupervised or self-supervised contrastive learning approaches (with implementations such as SimCLR), to a fully-supervised application. Unsupervised and self-supervised contrastive learning excels when there is a lack of sufficient labeled data, but supervised contrastive learning is able to make good use of labeled data when it is available (Khosla et al.).

3 Dataset and Features

As our dataset, we used the Clothing Dataset by Alexey Grigorev. This dataset consists of 5,403 labeled images of clothes. It comprises of thousands of crowdsourced images of clothing that have been submitted to form the dataset. The images were labeled as either of the following:

- T-Shirt (1011 items)
- Long Sleeve (699 items)
- Pants (692 items)
- Shoes (431 items)
- Shirt (378 items)
- Dress (357 items)
- Outwear (312 items)
- Shorts (308 items)
- Hat (171 items)
- Skirt (155 items)
- Polo (120 items)
- Undershirt (118 items)
- Blazer (109 items)
- Hoodie (100 items)

The dataset that we used had all the images in a single directory, unsorted. We first wrote some Python code to sort them into directories.

Since some of the rarer classes in the dataset only had a handful of images, we planned to train on classes with at least 100 images. This cut the dataset down to the top 14 classes, consisting of 5,268 labelled images. The images are of varying sizes and resolutions, so we scaled each of them down to a common, compressed resolution to expedite training. We normalized images by scaling pixel values by 1/255, corresponding to 128*128 pixels. Given the high imbalance and small number of

data in the number of images, we conducted data augmentation. We flipped the image horizontally and vertically, and did a random rotation between ±0.2 of a full rotation (e.g. 0.5 would correspond to 180° rotation).

After the data processing, we had 3600 training, 900 validation, and 461 test examples. Figure 1 shows some examples from the dataset.

Figure 1: Examples from the Dataset

4 Methods

In the study, we tried using 3 different algorithms. We first used a basic CNN model, then implemented transfer learning with ResNet-50, and Supervised Contrastive Learning.

4.1 Basic CNN

For our first attempt, we used a basic CNN for multi-class classification. CNNs are composed of convolutional layers, pooling layers, and fully-connected layers. We implemented our convolutional layer with 2 layers. These layers are responsible for extracting features from the input image, which are done by convolving the input with a set of learnable filters and detecting different visual patterns. We then had a pooling layer to downsample the data. This reduced the dimensions, making the model more efficient. For the fully-connected layer, we used the softmax for the activation function to make multi-class predictions based on the extracted features.

For our model, we started off with a normal tf.keras.Sequential() model with several Conv2D() and MaxPool() layers, followed by a Flatten(), Dense(), and output layer.

To train our CNN model, stochastic gradient descent was used to minimize the loss function. For our loss function, we used the Sparse Categorical Cross-entropy from Karas, rather than Categorical Cross-entropy, because our Y_i are not one-hot vectors but intergers of class indices. Using sparse categorical cross entropy saves time in memory and computation, as it just uses an integer instead of a whole vector. The loss function for the Sparse Categorical Cross-entropy is as follows: where w is the model parameters; y_i is the true label; $\hat{y}_i$ is the predicted label. Below are the hyperparameters used for the model.

$$J(w) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

```
num_train = 3600
num_validate = 900
num_test = 461
epochs = 50
```

```
batch_size = 16
img_shape = (128, 128, 3)
learning_rate = 0.001
optimizer: Adam
```

As the model was trained, the weights of the filters in the convolutional layers are adjusted to maximize the accuracy of the predictions.

4.2 ResNet-50

Since training deep learning models from scratch requires a large database of images and computational power, transfer learning has been an effective solution. Specifically, Residual learning allows us to solve the problems that arise when trying to train deeper neural networks, such as varnishing gradients and degradations. The Residual Network (ResNet) consists of stages with residual blocks with identity connections, where a layer is connected to the next layer and also to layers a few skips away. The ResNet-50 is a specific implementation of the ResNet model with 50 layers that has been pre-trained on the ImageNet dataset, containing millions of labeled images from 1000 different classes.

In transfer learning, only the classifier is trained in the network, while the features learned from the pretrained dataset are transferred. In our implementation, we implemented transfer learning with the ResNet50 network with the same Sparse Categorical Cross-entropy from Karas as the loss function.

We also have the same Flatten(), Dense(), and output layers as before, right after the ResNet50 network.

4.3 SupCon

Supervised Contrastive Learning (SupCon) is a CNN model that is designed for representation learning, which trains a model to learn representations of an image, helping it better classify those images later. For our SupCon model, we used a contrastive loss function that measures the similarity between the representations of different data points. The objective of this model is to minimize this loss so that the representations of similar data points are more similar to each other than the representations of dissimilar data points. This results in the model being able to push similar images closer together and dissimilar images further apart. In our model, we first trained an encoder that will help encode the features of an image. This way, images with similar features will be encoded similarly, and vice versa. For this encoder, we implemented transfer learning with the ResNet50V2 architecture, which we then continued to train. Then, we train a regular classifier on top of the pre-trained encoder, which includes a number of Dense layers.

We used the same hyperparameters from Method 4.1, with two additional hyperparameters:

```
temperature: 0.05
projection_units = 128
```

5 Results and Discussion

5.1 Basic CNN

Our basic CNN achieved a training accuracy of 77% but a test accuracy of just 25%, demonstrating a severe case of overfitting. For reference, the probability of randomly guessing a classification from 14 classes is about 7%, so this model is still better than a random guess. However, an accuracy of just 25% is extremely low and such a model would be unusable in the real world.

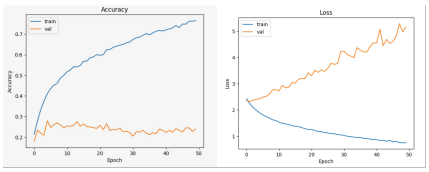

Figure 2: Train and Val Accuracy for Basic CNN

5.2 ResNet50

Our transfer learning approach with ResNet50 performed better, with a training accuracy of 91% and a test accuracy of 50%. Still, the model overfits, with the training accuracy being considerably higher than the test accuracy, but it is a vast improvement from our basic CNN. Our results here demonstrate not only the incredible potential of transfer learning to improve upon our own model, but also the lack of quality within our data, with a complex network like ResNet50 still unable to reliably find many trends within our dataset.

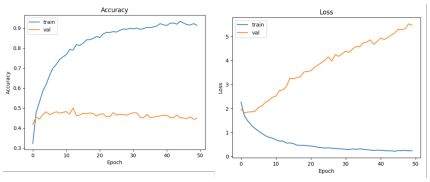

Figure 3: Train and Val Accuracy for ResNet50

5.3 SupCon

Our supervised contrastive learning model performed the best compared to our other methods, with a training accuracy of 81% and a test accuracy of 63%. It is apparent that the SupCon model still suffers from overfitting to the training data, but to a lesser extent than our other models. By learning some features of the images and grouping similar images together, it has been able to achieve higher accuracy than our transfer learning model.

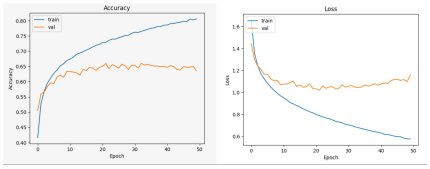

Figure 4: Train and Val Accuracy for SupCon

5.4 Discussion

Our results show that the transfer learning approach performed better than just a basic CNN, and that our supervised contrastive learning model performed better than the ResNet50-based approach. However, all three models suffered from overfitting to our training dataset to varying degrees.

Figure 5: Accuracy after 50 epochs

Before even working on the model, we spent quite a lot of time on just installing TensorFlow and other required libraries. We both have M1 Macbooks, and TensorFlow has many issues with Apple Silicon. TensorFlow was simply not in our Pip, even after many Pip reinstalls and Python downgrades. Even when it was there, the installed version would not import ("error: illegal hardware instruction"). Finally, we found an Apple fork of TensorFlow that, with much tinkering, finally seemed to work. In the future, working with a cloud computing service such as AWS or Google Cloud might be a reasonable workaround until maintainers of TensorFlow and other libraries improve compatibility with the relatively-recent Apple Silicon hardware.

The main limitation of our project was in the quality, or lack thereof, of our data. There were several major, glaring issues with our dataset. First, there were not very many images. With about 5,000 images in total, split among 14 classes, that makes for a pretty small sample size per class. Additionally, there was a large imbalance between the number of images per class. For instance, there were over 1,000 T-Shirt images, but only 100 Hoodie images. This definitely impacted the quality of our models, as having just 100 images for any given class is nowhere near enough to train a high-quality, robust classifier.

Furthermore, the images from within the dataset were crowdsourced from various social media campaigns and company sponsorships. The images were taken by the users themselves and submitted to be used in the dataset. This helps the dataset creators more easily acquire images for the dataset, but the downside is that user-submitted images are quite inconsistent and may have varying backgrounds, camera specifications, zoom, and picture angles. In a hypothetical laundry folding machine, if pictures of clothes were to be taken before folding, lighting, angles, and camera quality would be the same across any machine and any piece of clothing, which would help a model operate more effectively. The lack of standardization and a certain level of image quality, coupled with the relatively small size of our dataset, led to subpar performance of all of our models. For a similar project to be conducted in the future, a more extensive and higher-quality dataset should be used.

Other potential algorithms to be explored in the future include transfer learning with a variety of other types of pre-trained networks, such as MobileNet, VGG, or Inception. Additionally, algorithms from a different paradigm than the one we investigated, such as random forest classification, could have the potential for this task.

6 Conclusion

Our project attempted to classify images of clothes into different types, which would be applicable to a (yet-to-be-developed) laundry sorting machine. We explored three different methodologies to conduct this classification: a regular convolutional neural network approach, a transfer learning approach with ResNet50, and a supervised contrastive learning approach. The supervised contrastive learning method ended up being the most effective, while the other two methods trailed behind. Due to the poor quality of our dataset, our models were subject to overfitting our training set, and in the future, more should be done to procure higher quality and larger amounts of data.

7 Contributions

Andy worked on the database preprocessing and augmentation, research and architecture of the SupCon model, generating diagrams for the results section, and constructing code for the improved SupCon model. Hiro worked on researching literature on related works and different models, explanation of algorithms in methods, architecture and constructing code for the baseline model. We worked together on coding up the models, tuning hyperparameters, and discussing results.

8 References

8.1 Bibliography

Chen, Ting, et al. A Simple Framework for Contrastive Learning of Visual Representations. arXiv:2002.05709, arXiv, 30 June 2020. arXiv.org, http://arxiv.org/abs/2002.05709.

Ezat, W. A., et al. "Multi-Class Image Classification Using Deep Learning Algorithm." Journal of Physics: Conference Series, vol. 1447, no. 1, Jan. 2020, p. 012021. DOI.org (Crossref), https://doi.org/10.1088/1742-6596/1447/1/012021.

Fortune Business Insights. Washing Machine Market Size Share | Research Report [2028]. May 2021, https://www.fortunebusinessinsights.com/washing-machine-market-102645.

Global Industry Analysts, Inc. With Market Size Valued at $5.9 Billion by 2026, It's a Stable Outlook for the Global Commercial Laundry Machinery Market. 3 June 2022, https://www.prnewswire.com/news-releases/with-market-size-valued-at-5-9-billion-by-2026–its-a-stable-outlook-for-the-global-commercial-laundry-machinery-market-301557669.html.

He, Kaiming, et al. Deep Residual Learning for Image Recognition. Dec. 2015. arxiv.org, https://doi.org/10.48550/arXiv.1512.03385.

Jain, Rachna, et al. "Convolutional Neural Network Based Alzheimer's Disease Classification from Magnetic Resonance Brain Images." Cognitive Systems Research, vol. 57, Oct. 2019. ResearchGate, https://doi.org/10.1016/j.cogsys.2018.12.015.

Khosla, Prannay, et al. Supervised Contrastive Learning. arXiv:2004.11362, arXiv, 10 Mar. 2021. arXiv.org, http://arxiv.org/abs/2004.11362.

Krizhevsky, Alex, et al. "ImageNet Classification with Deep Convolutional Neural Networks." Advances in Neural Information Processing Systems, vol. 25, Curran Associates, 2012. https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html.

Lu, Siyuan, et al. "Pathological Brain Detection Based on AlexNet and Transfer Learning." Journal of Computational Science, vol. 30, Nov. 2018. ResearchGate, https://doi.org/10.1016/j.jocs.2018.11.008.

Reddy, A. Sai Bharadwaj, and D. Sujitha Juliet. "Transfer Learning with ResNet-50 for Malaria Cell-Image Classification." 2019 International Conference on Communication and Signal Processing (ICCSP), 2019, pp. 0945–49. IEEE Xplore, https://doi.org/10.1109/ICCSP.2019.8697909.

Singh, Inderpreet, et al. "AlexNet Architecture Based Convolutional Neural Network for Toxic Comments Classification." Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 9, Oct. 2022, pp. 7547–58. ScienceDirect, https://doi.org/10.1016/j.jksuci.2022.06.007.

Talo, Muhammed, et al. Convolutional Neural Networks for Multi-Class Brain Disease Detection Using MRI Images | Elsevier Enhanced Reader. https://doi.org/10.1016/j.compmedimag.2019.101673. Accessed 9 Dec. 2022.

8.2 Database and Libraries

Dataset URL: `https://github.com/alexeygrigorev/clothing-dataset`
Code URL: `https://github.com/andy0liang/laundry-sorting`