CNN Based Classification and Transfer Learning

Abdelkader Habel, Ali Abdel-Jalil, Oussama Cherifi, Zakaria El Manar El Bouanani

Abstract

This project explored the transfer learning abilities of Convolutional Neural Networks (CNNs) in image classification across both medical and general domains. Utilizing the ResNet-18 architecture, we developed two distinct models: one custom-trained on a colorectal cancer dataset and another leveraging pre-trained ImageNet1k weights. The research focused on colorectal and prostate cancer images, alongside an animal faces dataset, to assess the models' capacity in feature extraction and classification. The methodology includes preprocessing steps like resizing and normalization, followed by using the two ResNet-18 models for extracting features. Classification was performed using the K-Nearest Neighbors (KNN) and Support Vector Machine (SVM) algorithms. Additionally, t-SNE was applied for dimensional reduction and visualization of features. Results revealed that the custom-trained model performs well in classifying medical images, while the pre-trained model shows a broader classification strength, performing well in both medical and non-medical contexts. Notably, in the prostate cancer dataset, the pre-trained ImageNet model demonstrated that the SVM classifier was more effective than KNN, though both maintained high metrics in precision, recall, and F1-Score. This comprehensive approach highlights the different capabilities of CNNs in diverse image classification tasks.

1. Introduction

Computer vision is a field of computer science that focuses on extracting meaningful data from images and which often uses machine learning techniques as tools. This data can serve as input to models and help derive important features to make predictions about visual observations. Over the past decades, the field of deep learning has evolved to where models can train on medical images and detect pathologies, helping medical practitioners detect and treat patients early. One type of model is the Convolutional Neural Network (CNN), useful when the data is arranged in a grid shape, like images that can be seen as matrices of pixels. CNNs can handle large amounts of data and can extract features from images, without the need to manually do it. Furthermore, they can be retrained to perform different tasks on different sets of data, which is what we are aiming for in this project.

There are several challenges faced in the field of computer vision and deep learning. Some of these involve the limited availability of data, the limitations imposed by hardware as well as the difficult interpretability of models in crucial applications. In deep learning, the amount of data is crucial to train models properly to allow them to generalize well. However, it is not always easy to get it, especially in some niche fields, like pathologies or more domain specific fields. Even with a good amount of data, there is a hardware challenge that can be faced since CNNs are computationally hungry. Large amounts of resources are needed to train heavy models in timeframes that are acceptable. When it comes to the interpretability of models, it may be a huge drawback in fields like medicine, where it may be necessary to understand how a model makes a decision. The problem is that they are hard to understand for humans due to the representation of the data internally.

The project's goal is to observe the transfer learning capabilities of a pre-trained Convolutional Neural Network (CNN) alongside a custom model we train on a colorectal cancer dataset. We used 3 datasets, one about colorectal cancer, one about prostate cancer and one on animal faces. They will often be referenced as datasets 1, 2 and 3 respectively. In the process of solving the problem, which is feature extraction and classification on different domain datasets, several steps were taken. Firstly, all images were normalized prior to being manipulated. A ResNet-18 network was then trained on the colorectal cancer dataset and used to extract the features in all 3 datasets. The classification with the custom trained model on a test set of the colorectal cancer gave accuracies above 96%. In parallel, a ResNet-18 model trained on the ImageNet dataset was applied for feature extraction on the 2 last datasets. The 5 sets of extracted features were then visualized after applying dimensionality reduction technique t-SNE. The features extracted on the prostate dataset by the ResNet-18 model trained on ImageNet were then classified with the KNN and SVM algorithms.

Prior to conducting the experiment, the expectation was that the ImageNet model would perform better in feature extraction on the dataset of animal faces than that of prostate cancer. On the other hand, the custom CNN was expected to perform better on the prostate cancer dataset over that of the animals. Indeed, we expected similarities between features in different cancer types, whereas the features found in cells are not found in animal faces. The initial expectations were confirmed by the t-SNE plots who showed the separation of classes, where the custom CNN was incapable of clustering the animal faces but was able to cluster the prostate cancer images. On the other hand, the pretrained model from PyTorch exceeded expectations in its clustering of wild animals, going as far as making defined clusters for sub-categories in the class. It was also better at clustering the prostate cancer cells than the custom CNN.

Classification techniques KNN and SVM were used on the extracted features of the prostate cancer dataset with the pretrained model. The results were mostly positive, with the SVM model slightly outperforming the KNN model.

The SVM model had a near-perfect score for the precision, recall and F1-Score metrics whereas the KNN model's metrics were around 95%. A confusion matrix was produced and it was discovered the KNN model was prone to misattributing the Gland label to the two other classes. The SVM on the other hand was better at classifying the right classes, and lowered the errors for the gland class. The confusion matrix confirmed what was observed in the classification reports; the SVM was better at classifying features than the KNN algorithm.

Studies have demonstrated the efficacy of CNNs in accurately classifying various types of cancerous tissues from histological images [1]. This aligns with our use of colorectal and prostate cancer datasets.

In contrast, general image classification using CNNs, as seen in other studies, involves more diverse datasets like the Animal FacesHQ dataset used in our project. These studies have shown how pre-trained models like ResNet-18, trained on large and varied datasets like ImageNet, can be effectively adapted to new, unseen datasets through transfer learning [2].

2. Methodology

2.1. Datasets

In this project three datasets were used. The first is a collection of eight different types of colorectal cancer tissues [3]. Tissues were collected from patients at the National Center for Tumor Diseases and the University Medical Center Mannheim in Germany. The second dataset is a collection of histological slides of prostate cancer tissues split into three different classes [4, 5]. The dataset originates from various institutions and universities including: The Cancer Genome Atlas (TCGA), Case Western University, University Hospital Cologne and Hospital Wiener Neustadt. The Animal FacesHQ (AFHQ) dataset consists of animal faces of cat, dog, and wildlife [6]. It was introduced by Choi et al. in the context of their StarGAN v2 work. For the project we used 6000 images per dataset, spread across 3 classes per set. A summary of the statistics of each dataset is shown in Table 1.

Dataset:	Colorectal	Prostate	Animals
# images (original)	100 000	120 000	16 000
# images (project)	6 000	6 000	6 000
Classes (original)	8	3	3
Classes (reduced)	3	3	3
Format (pixels)	224x224	300x300	512x512

Table 1: Basic information on the datasets

The pre-processing step consisted of normalization using the mean and standard deviation values from the ImageNet dataset: mean=[0.485, 0.456, 0.406] and std=[0.229, 0.224, 0.225]. Using them serves two main purposes in this study. First, it ensures that the model trained from scratch is on the same level with the pre-trained model on ImageNet1k, as it aligns the data distribution with that of the ImageNet1k dataset. This similarity in data pre-processing makes the comparisons between both models more valid and meaningful. Secondly, employing these normalization values for the images tested with the pre-trained model enhances the reliability of our results. This approach ensures that the testing conditions closely mirror the conditions under which the pre-trained model was originally trained.

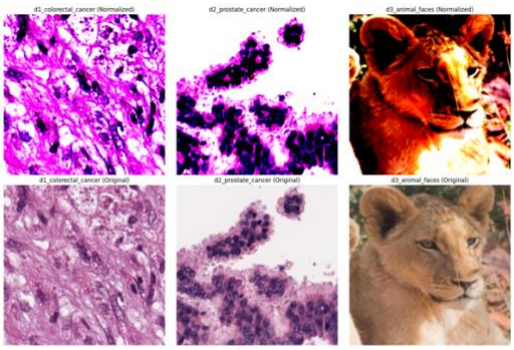

Figure 1: The top row shows the normalized image, and the second row shows its original version. 1 image randomly selected per dataset.

From the first dataset there are 3 classes to consider: Smooth muscle (MUS), normal colon mucosa (NORM) and cancer-associated stroma (STR). The MUS class appears to have a more uniform texture and pattern with a fiber-like composition. The NORM class has more variation in textures and some rounded spots that look like bubbles. We can also observe some black dots that also appear in the MUS class. The cancer-associated stroma class is extremely similar to the MUS class. It also has a fiber-like structure with uniform texture, but it seems that some images have more black dots and some invariability around the fibers. This resemblance suggests that the model may confuse MUS with STR during testing due to their visual similarities. All classes are equally represented,

with a balanced number of images for each class in the dataset.

Each dataset was split in a 7:3 ratio, with random shuffling applied to the training set to prevent overfitting, while the testing set was not shuffled. We did not explicitly use cross-fold validation in this setup; however, it could be considered for future iterations to enhance the model's generalizability. All images were resized to a resolution of 224x224 pixels to standardize the input, enhance computational efficiency and due to it being a conventional value used in ResNet models.

2.2 CNN-based classification

Task 1 can be generally summarized into 4 main methods: data preparation and preprocessing, model initialization and setup, training phase and evaluation phase.

The selected architecture for the task is ResNet18, a residual network with 18 layers. ResNet architectures are known for their ability to train deep networks by using skip connections or shortcuts to jump over some layer. These connections help to address vanishing gradient problems by allowing the gradient to flow through the network without diminishing.[7] This makes this architecture suitable for a wide range of image classification tasks, including those with complex image datasets. Furthermore, this architecture was chosen in this project due to it being understood well by the authors of this report.

The custom ResNet was initialized with no weights and was modified to suit the needs of the project. The fully connected layer is replaced with an identity layer from PyTorch, which essentially outputs the output of the previous layer [8], which are the features. In summary, the fully connected layer was removed to get the features only and not the predictions. Then, a prediction layer is made and contains three neurons for three classes. When the model is trained, it makes a forward pass using *model.fc_pred(input)* instead of *model(inputs)*. The difference is that the latter outputs features while the former outputs classification results.

The cross-entropy loss and stochastic gradient descent (SGD) optimizer were used in the training phase. A scheduler was also used to adapt the learning rate after a given number of epochs. Each epoch reports its loss and accuracy

The wall clock time of 26 seconds for an epoch suggests that the training process of the model is quite efficient. This quick training time is particularly beneficial in a development setting, where multiple iterations and adjustments to the model are common. In terms of computational complexity, the FLOPS measure for ResNet18, which is approximately 1.82 billion [9], indicates the number of floating-point operations the model performs to process a single input and generate an output. This level of FLOPS is relatively moderate, especially when compared to more complex models in the ResNet family, such as ResNet50 or ResNet101, which require a significantly higher number of FLOPS. This lower FLOPS count is indicative of ResNet18 being one of the smaller and less computationally intensive models in the ResNet series [10].

To visualize the extracted features, t-SNE reduction technique was necessary to plot the features in a 2-dimensional plane. It lets us spot patterns and relationships in the data that might not be obvious at first glance by dimensionally reducing the hyperspace of features. By also laying out the features into clusters, t-SNE can show us if the model is having a hard time telling some classes apart. It confirms what we've noticed before: some classes look similar to the model, as shown by how their clusters are close on the graph. This kind of insight is useful for tweaking the model to tell these classes apart better.

2.3 Feature extraction

In Task 2, the objective was to extract the features from datasets 2 and 3 using pre-trained CNN encoders. The extraction was performed with the previously developed model and with a pretrained ImageNet model. The extraction consisted of making a forward pass in the model, as discussed previously, and getting the outputs which are features and not class probabilities. With the fully connected layer being replaced with an identity layer, the output of the network is effectively a vector of features. The output is a 512xN vector, where N is the number of samples. The 512 value is obtained after several convolutions and pooling layers, documented in PyTorch's documentation [11].

KNN and SVM algorithms were used to classify the extracted features. KNN works by considering the distances between the data points which is calculated based on the features of the data points. For classification it determines the majority class among the K nearest neighbors in order to assign the data point, where K is 3 for the three classes. SVM was chosen as a second technique because it works effectively with non-linear relationships, as well as being well understood by the authors of the report.

3. Results

3.1 Experiment setup

The experiments were conducted with an SGD optimizer, a cross entropy loss and a learning rate scheduler. The model trained on a portion of the colorectal cancer dataset

for 10 epochs. The model evaluation and performance assessment happened by running our model on the testing set and calculating the test accuracy.

Hyperparameters selection was guided by online recommendations for ResNet18 [12]. The learning rate for SGD was set at 0.01, paired with an adjustment scheduler, which is a standard approach for SGD to manage the optimization process effectively. The scheduler has the responsibility of adapting the learning rate by a certain amount after a certain number of epochs [13]. We opted for a batch size of 32, offering a balance between computational efficiency and model learning speed. Values for the number of epochs were initially chosen to be between the range of 10 to 50. In the end, it was decided that 10 epochs is a suitable value, as a higher number of epochs did not significantly improve the performance for the considerable excess time it took to train.

The precision, recall and F1-Score were computed on the test set for the colorectal cancer, and the values were above 98% for all 3, indicating that the model can accurately classify the colorectal cancer.

With results being satisfying, visualization of the features after t-SNE reduction was conducted. The same extraction and visualization process was executed for datasets 2 and 3 for the two ResNet models. Finally, KNN and SVM classification methods were applied on the extracted features of the prostate cancer dataset with the ImageNet-trained model.

For the KNN implementation, K was set at 3, obviously expecting 3 clusters, one per class. In the case of SVM, dealing with non-linear data, an RBF kernel was used. These hyperparameter choices, including learning rate, momentum, epochs, and batch size, were iteratively refined to best fit our specific scenario.

3.2 Main results

During the training of the custom model, the accuracy started at high values, from the very beginning. It started around 75% accuracy and went to around 98% after 10 epochs.

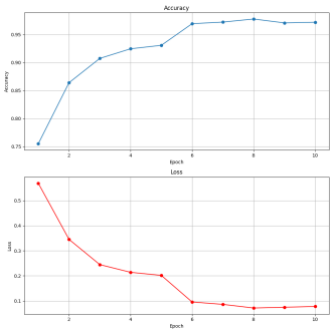

Figure 2: Accuracy and Loss per epoch

The accuracy graph shows a positive trend, with the model's accuracy improving from epoch 1 to epoch 10. This indicates that the model is learning from the training data and is becoming better at correctly classifying the input images over time. The plateauing of the curve towards the later epochs tells that the model may be approaching its peak performance on the given training dataset.

On the other hand, the loss graph which represents the model's error rate, shows a sharp decline initially, which calms off as the epochs progress. The declining loss is expected as the model optimizes its weights and biases to reduce the prediction error. The flattening of the loss curve suggests that the model is reaching a point where further learning on the training dataset will not significantly reduce the error rate.

With a test loss of 0.0532 and an accuracy of 98.33%, the model demonstrates high precision and a low error rate on the test dataset, indicating robust performance in classifying unseen data. Furthermore, the values for precision, recall and F1-Score were respectively 98.34%, 98.33% and 98.33%.

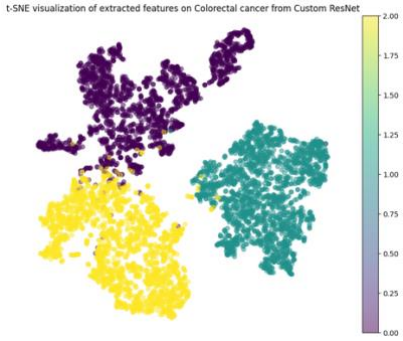

Figure 3: t-SNE visualization of extracted features on Colorectal cancer from Custom ResNet

The t-SNE visualization presented shows the feature distribution of three distinct classes obtained from the Custom ResNet model trained on colorectal cancer data. In this graph, we have Class 0 (Smooth Muscle, MUS) in

purple, Class 1 (Normal Colon Mucosa, NORM) in blue-green, and Class 2 (Cancer-Associated Stroma, STR) in yellow.

Analyzing Figure 3, it is observed there are 3 distinct clusters for each class, which indicates that the model has learned to extract features that are significantly different for each category. However, there are some overlaps between the purple (MUS) and yellow (STR) clusters. This overlap suggests that the model sometimes confuses features between these two classes. This observation makes sense since the images from MUS and STR are visually similar. In contrast, the blue green (NORM) cluster is more separated, implying that the features of the normal colon mucosa are distinct enough for the model to differentiate them more clearly from the other two.

The custom model, applied on datasets 2 and 3, resulted in the two following plots for prostate cancer and animal faces.

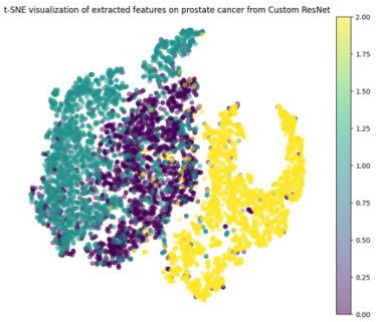

Figure 4: t-SNE visualization of extracted features on prostate cancer from Custom ResNet

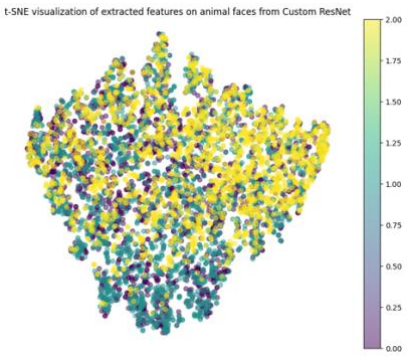

Figure 5: t-SNE visualization of extracted features on animal faces from Custom ResNet

It is seen that the model distinguishes between different classes in Figure 4, but with less accuracy than it did for the colorectal cancer dataset. Indeed, classes 0 and 1 merge into each other, whereas class 2 has a slightly better separation. On the other hand, the model is unable to separate per class on the animal faces dataset. All the classes are present in a single cluster.

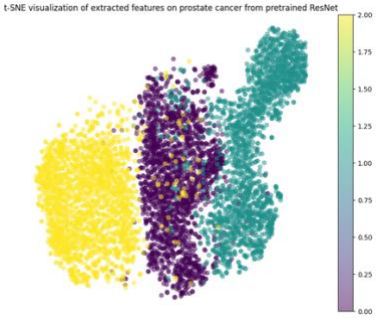

Figure 6: t-SNE visualization of extracted features on prostate cancer from pretrained ResNet

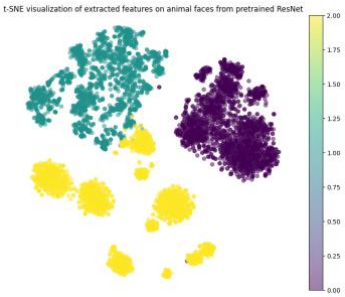

Figure 7: t-SNE visualization of extracted features on animal faces from pretrained ResNet

The pretrained model from PyTorch led to seeing a better feature extraction than the custom model. Indeed, Figure 5 shows the separation of classes, where although there is some mix-up between them, the separation is better than seen in Figure 3 with the custom model. In Figure 6, it can be seen that the model is able to almost perfectly separate the classes in its feature extraction. What can also be said about the extraction from the pretrained model on the set of animals is that within the wild animals cluster, there are sub-clusters, which can represent different categories in that class, like lion vs lioness, male or female wolf, tiger or fox. It is clear that the pretrained model is much better than the custom model, in both extractions.

KNN Classification Report:

	precision	recall	f1-score	support
0	0.88	0.97	0.92	597
1	0.98	0.93	0.95	604
2	0.99	0.94	0.97	599
accuracy			0.95	1800
macro avg	0.95	0.95	0.95	1800
weighted avg	0.95	0.95	0.95	1800

SVM Classification Report:

	precision	recall	f1-score	support
0	0.97	0.97	0.97	597
1	0.97	0.98	0.98	604
2	1.00	0.98	0.99	599
accuracy			0.98	1800
macro avg	0.98	0.98	0.98	1800
weighted avg	0.98	0.98	0.98	1800

Figure 8: Classification reports for KNN and SVM on the prostate cancer dataset with the pretrained model

Classification reports (Figure 8) generated for the KNN and SVM based classifications on the prostate cancer set showed good performance across all classes. Specifically,

the report for KNN had a particularly high recall for class 0 (Gland) and perfect precision for class 2 (Tumor). However, class 2 had a slightly lower recall, suggesting that while all predictions for class 2 are correct due to precision, the model missed some real class 2 samples.

The report for SVM shows excellent performance, with near-perfect scores in precision and recall for all classes, leading to very high F1-scores, a combination of precision and recall. The SVM model slightly outperformed the KNN model across all classes, particularly with class 0.

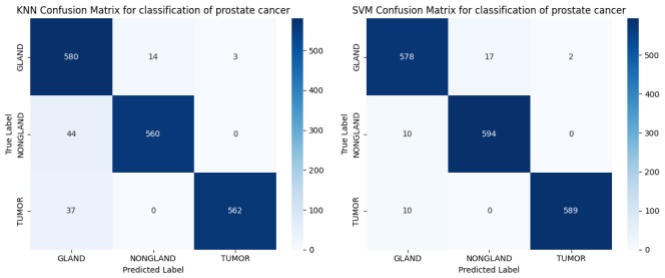

Figure 9: Confusion matrix for KNN and SVM on the prostate cancer dataset with the pretrained model

From the KNN confusion matrix (Figure 9), it can be said that the model generally performs well, but there are some glaring misclassifications between the classes. Specifically, 37 Tumor and 44 Nongland samples were incorrectly predicted as Gland.

In comparison, the SVM confusion matrix (Figure 9) shows a higher number of correct predictions and fewer misclassifications across all classes. The SVM model was better at predicting the Tumor class than the KNN was. The Gland class was misattributed to samples 20 times, compared to 81 with the KNN.

These heatmaps provide visual confirmation of the findings in the classification reports in that the SVM classifier performs better in distinguishing between the three classes, with higher accuracy and fewer errors. Notably, the SVM model was much better at predicting tumors and less prone to predicting Gland where it shouldn't.

References

[1] "Convolutional Neural Networks," Pathology Outlines - Convolutional neural networks, https://www.pathologyoutlines.com/topic/informaticsconvnet.html (accessed 2023).

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," arXiv.org, https://arxiv.org/abs/1512.03385 (accessed 2023).

[3] J. N. Kather, N. Halama, and A. Marx, "100,000 histological images of human colorectal cancer and healthy tissue," Zenodo, https://zenodo.org/records/1214456 (accessed 2023).

[4] Y. Tolkach, "Datasets digital pathology and artifacts, part 1," Zenodo, https://zenodo.org/record/4789576 (accessed 2023).

[5] Author links open overlay panelBirgid Schömig-Markiefka 1 et al., "Quality control stress test for deep learning-based diagnostic model in digital pathology," Modern Pathology, https://www.sciencedirect.com/science/article/pii/S0893395222003702?via%3Dihub (accessed 2023).

[6] Larxel, "Animal Faces," Kaggle, https://www.kaggle.com/datasets/andrewmvd/animal-faces (accessed 2023).

[7] "ResNet: Residual neural networks - easily explained!," Data Basecamp, https://databasecamp.de/en/ml/resnet-en (accessed 2023).

[8] "Identity," Identity - PyTorch 2.1 documentation, https://pytorch.org/docs/stable/generated/torch.nn.Identity.html (accessed 2023).

[9] "Model zoo," Model Zoo - MMClassification 0.25.0 documentation, https://mmclassification.readthedocs.io/en/latest/model_zoo.html (accessed 2023).

[10] Ladofa, ladofaladofa, "What is flops in field of deep learning?," Stack Overflow, https://stackoverflow.com/questions/58498651/what-is-flops-in-field-of-deep-learning (accessed 2023).

[11] "Source code for torchvision.models.resnet," torchvision.models.resnet - Torchvision main documentation, https://pytorch.org/vision/main/_modules/torchvision/models/resnet.html#resnet18 (accessed 2023).

[12] The hyper parameters choices for Resnet18 - Researchgate, https://www.researchgate.net/figure/The-hyper-parameters-choices-for-ResNet18_tbl2_339580876 (accessed 2023).

[13] S. Sudhakar, "Learning rate scheduler," Medium, https://towardsdatascience.com/learning-rate-scheduler-d8a55747dd90 (accessed 2023).