COMP 432 Final Report

Abstract

This project studies the use of Convolutional Neural Networks (CNNs) for image classification in computational pathology and computer vision. Three datasets are used: Colorectal Cancer, Prostate Cancer, and Animal Faces. The goal is to understand domain-specific challenges and explore how knowledge can transfer between these applications. In Task 1, a CNN is trained on the Colorectal Cancer dataset, and its performance is analyzed. The features extracted by the CNN are visualized using t-SNE. In Task 2, the CNN encoder from Task 1 is tested on the Prostate Cancer and Animal Faces datasets, and a pre-trained ImageNet CNN is also used for comparison. The extracted features are then analyzed with t-SNE and classified using traditional machine learning models. The results show how CNNs perform in different domains and how features extracted by a model trained in one domain can efficiently transfer to other similar models.

1. Introduction

The main objective of this project is to study the transferability of CNN models between different types of datasets. It is important to gain an understanding of transferability in machine learning as it allows for faster model training through and an efficient use of resources. Training machine learning models from scratch for every new task is computationally expensive and time-consuming. With pretrained weights, the performance of the CNN to accomplish a new classification task will be much quicker. This is because there are low-level features that are common with most computer vision tasks, such as shapes, edges, textures, and lines. Transfer learning offers the advantage of enabling a CNN to become a robust model even with limited data, as it leverages knowledge from other datasets. This capability is particularly impactful because it allows a single model to excel across multiple related tasks, fostering the discovery of new insights across disciplines.

The team noted three key issues. The first is the risk of overfitting, even if the data is large enough. This is because CNNs learn a large amount of features to complete the job. However, because they can learn so many features, they can also learn noise or redundancies that do not translate well to future datasets. The second challenge is domain shift, which arises when transferring a model between datasets with significantly different distributions, such as moving from Colorectal Cancer Classification to Animal Faces Classification. These types of errors can be corrected over time with extensive tuning and feature alignment, but there's no guarantee the model will ever be correct. The third challenge is the substantial computational resources required to train CNNs especially when handling large datasets for image classification. Training these models demands high-performance equipment and significant processing time, which may not always be available. The team's goal is to better understand CNN models and explore how transfer learning can be effectively applied to diverse-real-world scenarios.

This report tackles the challenge of transferability in CNN models by investigating how transfer learning can enhance performance across diverse datasets. To achieve this, a CNN model is first trained on a Colorectal Cancer Classification dataset, and then the model is reused and evaluated on Prostate Cancer Classification and Animal Faces Classification datasets. A portion of the Colorectal Cancer Classification dataset (Dataset 1) [2] contains approximately 6,000 images, evenly distributed across three types of colorectal cancer: smooth muscle (MUS), normal colon mucosa (NORM), and cancer-associated stroma (STR). All images are 224x224 pixels and are color-normalized using Macenko's method [1]. A portion of the Prostate Cancer Classification dataset (Dataset 2) [3] consists of approximately 6,000 images, equally divided among three types of prostate cancer: gland, nongland, and tumor. The images are 300x300 pixels in size. A portion of the Animal Faces Classification dataset (Dataset 3) [4] includes around 6,000 images, evenly distributed among three labels: cat, dog, and wild. This dataset, sourced from Kaggle, contains images that are 512x512 pixels.

The methodology to evaluate the effectiveness of transfer learning is split into two tasks. In task 1, the focus is on training a CNN model to perform a Colorectal Cancer Classification task. The model is trained on dataset 1, and the performance is evaluated through training accuracy and loss metrics. To gain a deeper insight into the learned feature representations, t-SNE [9] is utilized to reduce the dimensionality of the CNN encoder's output features and vi-

sualize them based on the corresponding class labels.

In task 2, the features extraction capabilities of the CNN encoder trained in task 1 and a pre-trained ImageNet CNN encoder on new datasets, Dataset 2 and 3 are investigated. Instead of training a new CNN model, the extracted features will be analyzed and visualized using t-SNE, creating visual representations that reveal how well the learned features generalize to different domains. For each dataset, a classical machine learning classifier will be used. In this case Support Vector Machines (SVM) [18] will be used to classify the extracted features.

The general implementation to complete task 1 was to load Dataset 1 and train a ResNet18 model [8] [15], from the PyTorch library from scratch. The model was adjusted to match the number of classes in the classification problem. The cross-entropy loss function and the Adam optimizer [17] were chosen for the model. The objective is to train a model and obtain high accuracy in classifying the inputs into the given categories: smooth muscle (MUS), normal colon mucosa (NORM), and cancer-associated stroma (STR). Another strategy used to increase the accuracy of the model is to perform a grid search [20] on a validation set to determine which hyperparameters (learning rate and batch size) work best for training the model to achieve high accuracy. Properly tuned hyperparameters help the model converge to an optimal solution.

The general implementation to complete task 2, the goal is to observe the transferability of a trained model on one problem domain to another. In task 2, the encoder obtained from the neural network pre-trained with colorectal cancer data will be tested on a problem domain similar to the training dataset (Prostate Cancer Classification) and on a vastly different problem domain to the training dataset (Animal Faces Classification).Using t-SNE, the extracted features can be analyzed and visualized according to their class labels to assess the accuracy and effectiveness of transferring knowledge from one classification task to another. The extracted features are then classified using an SVM classifier to determine how well the extracted features represent the data for distinguishing between classes.

To determine if the model performs at a high accuracy. The training model will go through an evaluation phase using a test dataset from Dataset 1 (20% of Dataset 1). The evaluation will compare the model's output to the ground truth label and determine the final accuracy of the model. It is expected for the model to obtain high accuracy as ResNet18 is a proven deep residual learning model for image recognition.

In task 2, the team anticipates that the model trained on colorectal cancer will perform poorly on Dataset 3 (Animal Face Classification) due to the differing problem domains. However, the team expects good performance on Dataset 2 (Prostate Cancer Classification), based on the hypothesis that the problem domains are similar. Furthermore, the team expects the pre-trained model on ImageNet to perform better, as it provides a larger and more diverse dataset enabling the model to learn from a wider range of features. This broader knowledge base will ultimately benefit the model when classifying both Dataset 2 and Dataset 3. These hypotheses can be confirmed by analyzing and visualizing the different t-SNE figures after applying the encoders from task 1 and ImageNet on Dataset 2 and Dataset 3. Performing a classification of the extracted features using SVM allows for determining whether a specialized encoder or a more generalized encoder, such as ImageNet, is more effective for transfer learning applications.

During the training of the vanilla ResNet18 model, one of the main problems was the speed at which the model was being trained on. Since the Dataset 1 contains 6000 images, the training process is due to take a long time if the task were to be completed entirely on CPU. To prevent this from happening, the implemented code should be able to run on GPU which will accelerate the training time by a considerable amount of time. A key challenge in task 2 is determining whether the model can perform well on a completely different domain than the one it was originally trained on. In task 2, a potential solution to improve transferability is to use a model pre-trained on the ImageNet dataset, which includes over 1,000 classes and millions of images. This approach may lead to higher accuracy when evaluated across two different datasets and classification tasks

1.1. Related works

The article written by A. Sharma [5] and S. Likhita [6] discusses the use of CNN for solving medical image classification problems for cancer detection, which is similar to the problem domain of the project. The article written by Sharma evaluates different CNN architectures such as: Sequential, Vgg 16, ResNet, and InceptionNet for predicting lung cancer from chest X-ray images. Among these models, ResNet was shown to be the most effective model achieving the highest accuracy due to its robust ability to capture critical image features. This study shows the potential of CNNs to handle complex medical imaging tasks and highlights the importance of selecting architectures that excel in feature extraction and classification. Similarly, the second article compares CNN and Support Vector Machine on the classification task for skin cancer detection. The results of this article is that CNNs outperformed SVMs with high accuracy and superior specificity, demonstrating the capability of CNNs to process intricate patterns in image data for effective diagnosis. The article written by E. Tri Hastuti [7] will provide background and context about transfer learning and the effectiveness of utilizing pre-trained neural networks. The study utilizes DenseNet121, a CNN architecture pre-trained on ImageNet, to leverage its feature ex-

traction capabilities for the new classification task. Transfer learning demonstrated superior performance compared to traditional learning achieving a higher accuracy. The DenseNet121 model's dense connections allowed efficient feature reuse, improving gradient flow and enabling accurate classification with fewer parameters. This research relates to this project as it highlights how pre-trained CNN models can adequately generalize learned features from a source domain, such as ImageNet to, to diverse target domains with proper fine-tuning.

2. Methodology

To address the colorectal cancer classification problem in task 1, ResNet18 CNN [8] [15] architecture was selected due to its effectiveness in image classification tasks and its ability to handle complex patterns in medical images. The model employs shortcut connections to directly connect lower layers to upper layers, helping mitigate the effect of vanishing gradients, a common issue when training deep neural networks. These shortcut connections enable the network to learn residual functions, enhancing its ability to capture fine-grained details and improving overall performance on complex datasets.

To further improve performance and mitigate overfitting, several enhancements were implemented. Data augmentation techniques were applied, such as horizontal flips, random rotations, and color normalization using Macenko's [1] method. These techniques artificially expand the dataset and introduce variability, making the model more robust to changes in input data. Additionally, t-SNE visualization was utilized to reduce the dimensionality of the learned feature space, enabling the identification of distinct clusters and providing insights into how well the model differentiates between classes. Grad-CAM was also employed to highlight regions of the images that influenced the model's predictions. This interpretability tool is particularly useful in medical applications, as it allows researchers to validate whether the model is focusing on clinically relevant areas, such as specific tissue structures or abnormalities.

Moreover, a grid search was performed to optimize hyperparameters such as learning rate and batch size, ensuring the model achieved optimal convergence during training. By combining these strategies, the ResNet 18 architecture was fine-tuned to effectively classify colorectal cancer images into distinct categories, demonstrating the potential of deep learning models in addressing challenging medical imaging problems.

2.1. Data Preparation for Task 1

The dataset consists of approximately 6000 images of colorectal cancer that can be classified into three classes: smooth muscle (MUS), normal colon mucosa (NORM) and cancer-associated stroma (STR). Each image was resized to 224x224 pixels to match the input dimensions of the ResNet-18 model. Furthermore, color normalization was applied to the images using Macenko's method, which is particularly helpful in this classification task as it standardizes stain intensity and removes color variability [1]. To increase the robustness of the model and reduce the chances of overfitting, data augmentation techniques were applied during training. Random rotations were applied to images to simulate different imaging angles horizontal flips were applied to introduce positional variance. The dataset was split into three sets: training set (60%), validation set (20%) and testing set (20%).

2.1.1 Data Training

Training was done using hyperparameter search (grid search) to evaluate different combinations of hyperparameter values generated by the grid search. It was specifically only applied to the validation set and not the training set or the testing set. This is because the purpose of the training set is to learn the parameters like weights and biases that would be used in the model. The parameters are then adjusted to minimize loss on the data. If grid search were to be applied on the training dataset, it would lead to the model overfitting the data as it will be optimizing its performance to fit data it will have already seen rather than on new unseen data. Similarly, grid search isn't used on the testing dataset because it would defy its purpose. The purpose of the testing dataset is to provide a realistic and unbiased measure of the model's ability to generalize on new unseen data. If grid search was used on the testing set, it would mean that the model would have already "seen" the testing data indirectly, making the final evaluation unreliable as the evaluation would be biased, thus invalidating the test results. The purpose of the grid search is to hyperparameter tuning and the validation set provides the perfect environment to test the different hyperparameter combinations without influencing the training or test performance. After training the model on the training set using various combinations of hyperparameters, the combinations were evaluated on the validation set and the model's performance was assessed. This helped in determining which combination of hyperparameters perform the best and should be used to improve the model's generalization. Once the model has been finalized, the test set was used to evaluate the model's performance and provide an unbiased estimate of how well the model performed on new, unseen data. Categorical cross entropy was used as the loss function because of its ability to handle multi-class classification by penalizing incorrect predictions based on their probability scores. The final fully connected layer of the ResNet-18 model was replaced with a dense layer with 3 output neurons, corresponding to the three output classes. These neurons are activated by a

softmax function for multi-class classification. The Adam optimizer was selected as the optimizer due to its adaptive learning rate capabilities. It was initialized with a learning rate of 0.001 to allow the model to make smaller adjustments to avoid overshooting and help it converge to a better local minima [21]. For training, a batch size of 32 was used to ensure stable updates and computational efficiency.

For the second task, we first had to finalize our CNN encoder created in task 1. Once we were satisfied with the results, we started preparing the data to be analyzed.

For the second task, it was necessary to finalize the CNN encoder developed during task 1 before proceeding. This involved thoroughly evaluating its performance on the initial dataset to ensure its reliability and accuracy. Metrics such as precision, recall, and F1-score were analyzed, along with visualizations generated through techniques like t-SNE and Grad-CAM, to confirm that the encoder had effectively learned meaningful features from the Colorectal Cancer dataset. Once the encoder's performance met the desired criteria, preparation for the next phase of data analysis commenced. This step involved pre-processing the new datasets, ensuring consistency in input dimensions, normalizing the data to improve model performance, and loading it into a Data Loader for efficient batch processing. These preparations laid the foundation for the subsequent feature extraction and comparative analysis with other encoders, marking the transition to task 2.

2.2. Data Preparation for Task 2

The second dataset consists of 120,000 images divided among three different classes: prostate cancer tumor tissue, benign glandular prostate tissue, and benign non-glandular prostate tissue. Due to computational and resource constraints, this dataset was scaled down by 95%, resulting in a subset of 6,000 images being used for the project. This downsizing maintained a balanced distribution across the three classes to ensure the integrity of the classification task.

The third dataset consists of 16,000 images categorized into three distinct classes: Cats, Dogs, and wildlife animals. Similar to the second dataset, the data was scaled down to 6,000 images, representing an even distribution across the three classes. This reduction was necessary to align with the project's processing limitations while retaining sufficient variability to test the encoders effectively.

Both datasets underwent preprocessing steps to transform the inputs into a uniform size of 224x224 pixels, suitable for the CNN encoder. Normalization techniques were applied to improve consistency and enhance the model's performance and accuracy. The preprocessed datasets were then loaded into a DataLoader to facilitate efficient iteration and batch processing. This setup was critical for managing the datasets during feature extraction and subsequent analysis.

Feature extraction was conducted using the CNN encoder developed in task 1. The encoder processed the datasets to generate features and corresponding labels, which were then used for further analysis. In addition, a pre-trained ResNet-18 encoder, with its fully connected layer removed, was utilized to extract features from both datasets, enabling a comparison of feature representations across the two encoders. The extracted features and labels were processed through a t-SNE model to reduce dimensionality and visualize the data. These visualizations provided valuable insights into the clustering and separability of the features generated by each encoder.

Following the visualization phase, the extracted features were evaluated using two classical machine learning models: Random Forest and SVM classifiers. The performance of these models was assessed based on precision, recall, and F1-scores. The SVM model consistently outperformed the Random Forest model across all metrics in the classification report, demonstrating superior capability in handling the extracted features. Consequently, the SVM model was selected for the final part of task 2.

The SVM model was subsequently employed to classify four distinct sets of features: two sets derived from the CNN encoder and two from the ImageNet encoder. These features were processed through the t-SNE model to generate comprehensive visualizations. The resulting graphs highlighted the effectiveness of the feature extraction methods and provided a detailed comparison of the encoders' ability to represent and classify data accurately. This analysis reinforced the strengths of the SVM model and offered insights into the relative performance of the encoders in handling diverse datasets.

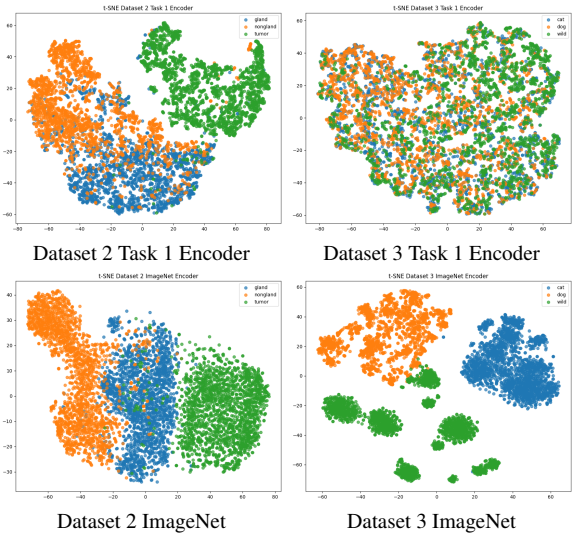

Figure 1. t-SNE plots evaluated on Dataset 2 and Dataset 3 by Task1 and ImageNet encoder

After receiving these results, we tested out a RandomForest classification model and an SVM classification model to see which performed better. The SVM model performed with a higher score in every way for the classification report than the RandomForest, so we decided to use it for the final part of task 2.

The SVM model was then used to classify the four different features extracted prior (two from our CNN encoder and two from the ImageNet encoder). These features were processed through the t-SNE model to produce the following graphs:

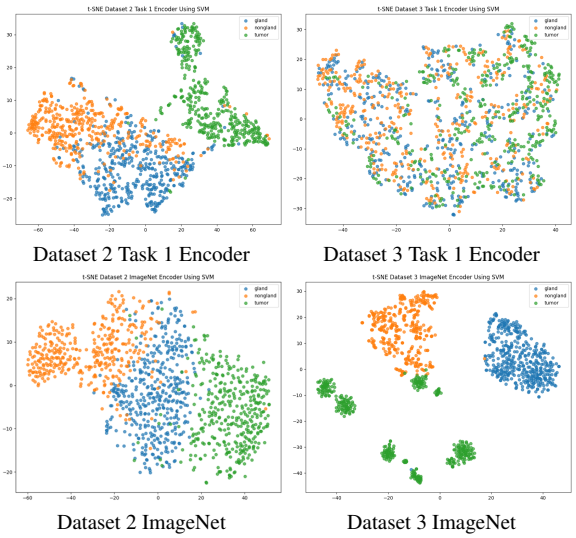

Figure 2. t-SNE plots evaluated on Dataset 2 and Dataset 3 by Task1 and ImageNet encoder and having SVM applied

3. Results

The model training starts after preprocessing the raw data by making some transformations. These include the normalization, resizing (224 x 224 pixels), and augmentation (horizontal flips) of the data. The data was divided into 3 parts: training (60%), validation (20%), and testing (20%). In order to train the model, a CNN (ResNet18) was chosen to capture the complex features of the images without overwhelming the other available resources.

The training ran on a batch size of 32, which may be slower but allows the model to retain the finer details of the different images and improve its classifying abilities. ResNet18's default output layer has been replaced with a fully connected (FC) layer using a softmax activation function to match the number of classes in the colorectal cancer dataset.

On the first attempt of training the model over 10 epochs, the training time took over 160 mins as it was running on CPU. As expected, it would take a significant amount of time to train a model over 6K images on CPU. By modifying the implementation of the code and some configurations, the team was able to train the model on GPU and it took less than 10 mins to fully train the neural network. This allowed more freedom for the team to manipulate and test the model. By saving a significant amount of time during the training process, the team was able to perform a grid search to determine which hyperparameters were most suited for the model [20].

Training ran on GPU over 10 epochs, with the Adam optimizer [17] and Cross-Entropy Loss [19]. These help in handling the varying magnitudes and directions of different gradients. It is also known to have a fast convergence, which helps in reducing the training time significantly [16]. For both the training and validation dataset, the accuracy and loss per epoch were tracked to ensure the model was not simply overfitting, but actually improving. By using grid search, it finds the right hyperparameters by trying different learning rates (0.1 to 0.0001) and batch sizes (8, 16, 32, 64). The best combo of hyperparameters has been chosen (learning rate: 0.001, batch size: 8), based on the validation set accuracy results (best validation accuracy: 95.25%).

The training loss over the epochs displays a steady decrease, indicating the model is fitting the training data better with each epoch (starts at 0.6021 and ends at 0.1877). Similarly, the training accuracy improves steadily across epochs, going from 74.17% at epoch 1 to 92.77 at epoch 10.

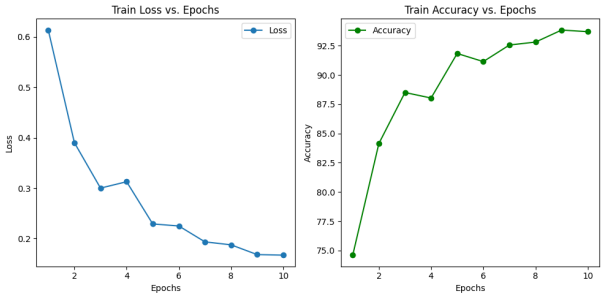

Figure 3. Loss and Accuracy VS Epochs for Training Dataset from Colorectal Cancer Dataset

On the other hand, the validation loss is not as consistent as the training loss. It begins at a relatively high value of 1.310 in epoch 1 and steadily declines to 0.1530 by epoch 10, indicating that the model is learning to generalize better to unseen data as training progresses. However, an anomaly is observed at epochs 5 and 6, where the validation loss unexpectedly increases to 0.680 instead of continuing its downward trend. This suggests that the model might momentarily struggle with generalizing to the validation data during these epochs.

The validation accuracy also exhibits fluctuations during training. It starts at 56.17% in epoch 1, indicating the

model's initial difficulty in correctly classifying the validation data. By epoch 4, the accuracy significantly improves to 89.33%, showcasing the effectiveness of the training process in the earlier stages. However, similar to the validation loss, a drop in accuracy is observed at epoch 5, where it decreases to 76.8%, before recovering to 93.76% by epoch 10. This fluctuation between epochs 5 and 6 could be attributed to potential overfitting, where the model starts to overly rely on features specific to the training data rather than general patterns that apply to new data. Another plausible explanation could be an imbalance in the validation dataset, where certain classes are underrepresented, causing the model to misclassify samples from those classes during certain epochs.

These irregularities highlight the importance of monitoring both validation loss and accuracy during training, as they provide critical insights into the model's generalization ability. Addressing these issues could involve incorporating additional data augmentation techniques to improve robustness or balancing the class distribution in the validation dataset. Additionally, fine-tuning hyperparameters, such as learning rate and regularization, might help mitigate these fluctuations and ensure more stable validation performance over the course of training.

Figure 4. Loss and Accuracy VS Epochs for Validation Dataset from Colorectal Cancer Dataset

By generating the classification report [12] based on the test dataset, it is possible to assess the overall performance of the model. The classification model achieved an accuracy of 95%, signifying that 95% of the predictions matched the actual class labels. This high level of accuracy reflects the model's capability to generalize well across the dataset and suggests that the learned features effectively captured the key characteristics of each class. For the MUS (smooth muscle) class, the model demonstrated a precision of 0.97, a recall of 0.92, and an F1-score of 0.94. These metrics indicate that the model was highly reliable in identifying MUS instances while maintaining a strong balance between precision and recall, even if a small percentage of MUS instances were misclassified.

For the NORM (normal colon mucosa) class, the model achieved its highest performance with a precision of 0.98, a recall of 0.99, and an F1-score of 0.99. This exceptional result demonstrates the model's near-perfect ability to correctly classify NORM instances while minimizing false positives and false negatives. This class's consistently high scores underline its distinctive feature representations, which likely contributed to the ease of accurate classification.

For the STR (cancer-associated stroma) class, the model achieved a precision of 0.91, a recall of 0.96, and an F1-score of 0.94. Although slightly lower than the other classes, these scores still represent robust performance, particularly given the complexity of differentiating cancer-associated stroma from other tissue types. The slight drop in precision suggests that a small number of non-STR instances were misclassified as STR, but the high recall indicates the model's effectiveness in identifying most STR instances within the dataset.

Overall, the results highlight the model's strong performance in differentiating between the MUS, NORM, and STR classes. The high F1-scores across all classes affirm the balance between precision and recall, making this model a reliable tool for classification tasks in this domain.

Classification Report:

	precision	recall	f1-score	support
MUS	0.97	0.91	0.94	400
NORM	0.98	0.99	0.99	401
STR	0.91	0.96	0.94	399
accuracy			0.95	1200
macro avg	0.95	0.95	0.95	1200
weighted avg	0.95	0.95	0.95	1200

Accuracy: 0.95

Figure 5. Classification Report on Test Dataset from Colorectal Cancer Dataset

To better represent how well the model's features represented each class, the test dataset was displayed in a t-SNE [9], projecting the high-dimensional features into a 2D space. Then, by working in conjunction with the t-SNE, we can compute the Silhouette Score [13], and quantify how well a data point fits into its assigned cluster and how distinct it is from other clusters. A Silhouette Score varies between -1 and 1, with values closer to 1 showing more distinct clusters. In our case, the score is estimated at 0.433 which shows reasonable structure is found.

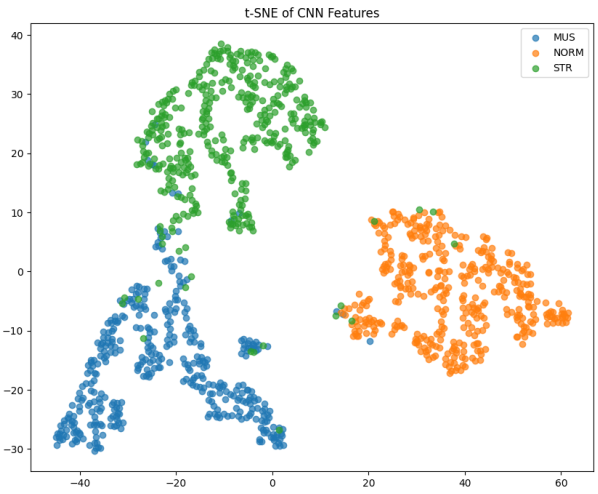

Figure 6. t-SNE plot of CNN Features on Test Dataset from Colorectal Cancer Dataset

A useful qualitative method to analyze our model's performance is the use of feature maps [10]. This CNN visualization technique applies filters to inputs and captures the results. Then, by using Grad-CAM [11], it can identify the sections of the image the ML model is focusing on. This can be useful, since the model could be focusing on an unimportant part of the image.

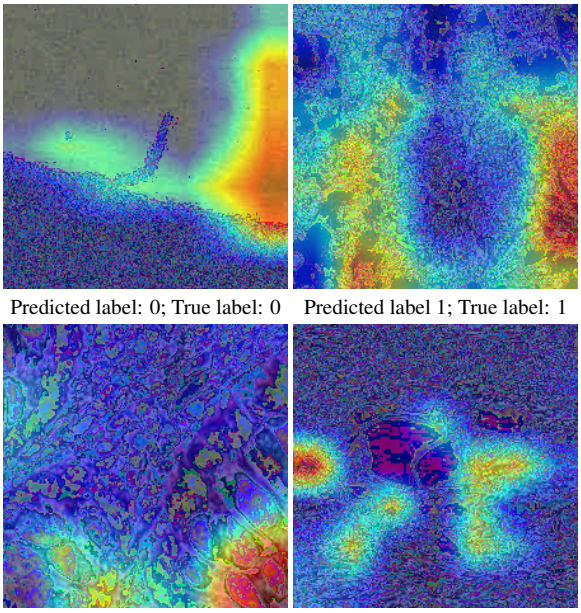

Figure 7. Feature maps on Grad-CAM on Test Dataset from Colorectal Cancer Dataset

The images shown are samples of application of feature maps and Grad-CAM, where labels 0, 1, and 2 correspond to smooth muscle (MUS), normal colon mucosa (NORM), and cancer-associated stroma (STR) respectively.

The setup for task 2 involved extracting the features from the prostate cancer dataset and the animal face dataset. This was first achieved using the CNN encoder developed in Task 1. Subsequently, another encoder was utilized, which was a pre-trained ImageNet encoder. Using these encoders, two versions of the features/labels for each dataset were created. This approach enables a comparison of which encoder classifies the data more effectively. After being processed through a t-SNE model, the results were graphed (see Figure 1).

From examining Figure 1, it can be concluded that the task one encoder performs reasonably well in classifying the different prostate cancer classes, as each cluster is distinctly separate. This indicates that the features extracted by the task one encoder are well-suited for datasets with characteristics similar to its training domain, such as medical imagery involving colorectal and prostate cancers. The ability to distinguish these classes highlights the encoder's capacity to generalize within related medical imaging tasks. However, the same level of performance is not observed for the animal face dataset, where the encoder struggles to accurately separate the data. The clusters for animal faces are less distinct, suggesting that the features learned from colorectal cancer images lack relevance to the visual patterns present in animal faces, such as fur texture, shapes, and complex backgrounds. ImageNet, as expected, demonstrates a high level of accuracy in classifying both datasets. The superior performance of ImageNet can be attributed to its training on a vast and diverse collection of images, encompassing numerous domains and object categories. This extensive training equips the ImageNet encoder with a broader understanding of generalizable features, enabling it to excel across varying datasets. This contrast underscores the significance of training data diversity in achieving robust and versatile feature extraction capabilities.

Subsequently, these same features were classified using an SVM model as the classical machine learning technique. The graphs generated after processing the new classifications through the t-SNE model are shown in Figure 2. The SVM model was chosen for its capability to effectively separate data points in high-dimensional space, leveraging the extracted features to produce meaningful decision boundaries. By utilizing both the task 1 encoder and the ImageNet encoder, the classification results were analyzed to assess the effectiveness of these feature representations. The classification reports provided precision, recall, and F1-scores for each dataset, allowing a detailed comparison of encoder performance. These metrics highlighted that while the ImageNet encoder consistently outperformed across tasks, the task 1 encoder showed strong performance within its spe-

cialized domain.

Dataset 2 (Task 1 Encoder) - SVM Results
Accuracy: 0.9391666666666667
Classification Report:

	precision	recall	f1-score	support
0	0.93	0.91	0.92	437
1	0.90	0.93	0.92	380
2	1.00	0.98	0.99	383
accuracy			0.94	1200
macro avg	0.94	0.94	0.94	1200
weighted avg	0.94	0.94	0.94	1200

Dataset 2 with Task 1 Encoder

Dataset 3 (Task 1 Encoder) - SVM Results
Accuracy: 0.7241666666666666
Classification Report:

	precision	recall	f1-score	support
0	0.70	0.66	0.68	437
1	0.70	0.73	0.72	380
2	0.77	0.79	0.78	383
accuracy			0.72	1200
macro avg	0.72	0.73	0.73	1200
weighted avg	0.72	0.72	0.72	1200

Dataset 3 with Task 1 Encoder

Dataset 2 (ImageNet Encoder) - SVM Results
Accuracy: 0.9708333333333333
Classification Report:

	precision	recall	f1-score	support
0	0.97	0.96	0.96	437
1	0.96	0.97	0.97	380
2	0.99	0.99	0.99	383
accuracy			0.97	1200
macro avg	0.97	0.97	0.97	1200
weighted avg	0.97	0.97	0.97	1200

Dataset 2 with ImageNet Encoder

Dataset 3 (ImageNet Encoder) - SVM Results
Accuracy: 0.9933333333333333
Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	437
1	0.99	0.99	0.99	380
2	0.99	0.99	0.99	383
accuracy			0.99	1200
macro avg	0.99	0.99	0.99	1200
weighted avg	0.99	0.99	0.99	1200

Dataset 3 with ImageNet Encoder

Figure 8. Classification of t-SNE SVM Classification Reports

The encoders were evaluated using the classification report from PyTorch which generates the precision, recall, and F1-score of a given model. These metrics provide a comprehensive evaluation of the model's performance, capturing its ability to correctly classify instances while balancing both false positives and false negatives. As explained prior, these scores are highly valuable for assessing the overall strength and reliability of the model.

The results indicate that the ImageNet encoder was overall better even when using the SVM classifier. This superior performance underscores the value of training on large, diverse datasets, which allows the encoder to extract features that are highly generalizable across domains. However, the task 1 encoder performed well in classifying the prostate cancer images, showcasing its capability to transfer knowledge within related medical domains. This outcome aligns with expectations, as the task 1 encoder was trained on images of colorectal cancer, a dataset that shares visual and structural similarities with the prostate cancer dataset. The shared domain features, such as tissue textures and patterns, likely contributed to its effective performance in this classification task.

In contrast, the task 1 encoder faced significant challenges in classifying the animal faces, reflecting the limitations of transfer learning when applied to vastly different domains. The encoder struggled to distinguish between classes in the animal face dataset, likely due to the absence of relevant visual features in its training data. Characteristics such as fur textures, varying facial structures, and non-uniform backgrounds present in the animal face dataset were beyond the scope of its learned features. Conversely, the ImageNet encoder successfully classified the animal faces with high accuracy, demonstrating the benefits of its extensive pre-training on a broad and diverse collection of images. The wide-ranging categories within the ImageNet dataset equip the encoder with the ability to generalize across distinct datasets, even those with significant domain shifts.

These results highlight the importance of dataset diversity in training models that are robust and versatile. While domain-specific encoders can excel within their trained scope, pre-trained models like ImageNet showcase the advantages of broad generalization, making them more effective for applications that span multiple, unrelated domains. This comparison reinforces the critical role of dataset composition and size in determining a model's adaptability and success in varied real-world tasks.

This study demonstrates the efficacy and limitations of Convolutional Neural Networks in domain-specific image classification and transfer learning applications. By leveraging a CNN encoder trained on a Colorectal Cancer dataset, the research highlights the potential for feature reuse in similar domains, such as Prostate Cancer classification, and the challenges faced when applied to vastly different datasets, like Animal Faces classification. The results underscore the advantages of using pre-trained models, such as those from ImageNet, which exhibit superior generalization capabilities due to exposure to diverse datasets.

The findings emphasize the importance of dataset similarity in transfer learning, as evidenced by the strong performance of the Colorectal Cancer encoder on Prostate Cancer data compared to its struggles with Animal Faces. Moreover, the integration of classical machine learning methods, like SVMs, for feature classification further reinforced the value of robust feature extraction in achieving high accuracy across tasks.

References

[1] Marc Macenko, *A method for normalizing histology slides for quantitative analysis*, IEEE Xplore, 2009. Available at: https://ieeexplore.ieee.org/document/5193250 (Accessed: December 8 2024). 1, 3

[2] Jakob Nikolas Kather, *100,000 histological images of human colorectal cancer and healthy tissue*, Zenodo, 2018. Available at: https://zenodo.org/records/1214456 (Accessed: December 8 2024). 1

[3] Yuri Tolkach, *Datasets Digital Pathology and Artifacts, Part 1*, Zenodo, 2021. Available at: https://zenodo.org/records/4789576 (Accessed: December 8 2024). 1

[4] Andrew Maranhão, *Animal Faces*, Kaggle, 2019. Available at: https://www.kaggle.com/datasets/andrewmvd/animal-faces (Accessed: December 8 2024). 1

[5] Ankita Sharma, *Comparative Analysis of Various CNN Models for Lung Cancer Prediction*, IEEE Xplore, 2024. Available at: https://ieeexplore-ieee-org.lib-ezproxy.concordia.ca/document/10511872 (Accessed: September 25 2024). 2

[6] S. Likhitha, *Skin Cancer Classification using CNN in Comparison with Support Vector Machine for Better Accuracy*, IEEE Xplore, 2022. Available at: https://ieeexplore-ieee-org.lib-ezproxy.concordia.ca/document/10047280 (Accessed: September 25 2024). 2

[7] Endang Tri Hastuti, *Performance of True Transfer Learning using CNN DenseNet121 for COVID-19 Detection from Chest X-Ray Images*, IEEE Xplore, 2021. Available at: https://ieeexplore-ieee-org.lib-ezproxy.concordia.ca/document/9537261 (Accessed: September 25 2024). 2

[8] Great Learning Editorial Team, *ResNet: What is Residual Neural Network?*, Great Learning, 2022. Available at: https://www.mygreatlearning.com/blog/resnet/ (Accessed: 25 September 2024). 2, 3

[9] Scikit-learn, *TSNE*, Scikit-learn developers, 2024. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html (Accessed: September 25 2024). 1, 6

[10] Arpit Jain, *Guide to Visualize Filters and Feature Maps in CNN*, Kaggle, 2019. Available at: https://www.kaggle.com/code/arpitjain007/guide-to-visualize-filters-and-feature-maps-in-cnn (Accessed: September 25 2024). 7

[11] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh and D. Batra, *Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization*, IEEE Xplore, 2017. Available at: https://ieeexplore.ieee.org/document/8237336 (Accessed: September 25 2024). 7

[12] Machine Learning, *Classification: Accuracy, recall, precision, and related metrics*, Google for Developers. Available at: https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall (Accessed: September 25). 6

[13] Scikit-learn, *silhouette score*, Scikit-learn developers, 2024. Available at: https://scikit-learn.org/1.5/modules/generated/sklearn.metrics.silhouette_score.html (Accessed: September 25 2024). 6

[14] Mathworks developers, *resnet18*, Mathworks, 2024. Available at: https://www.mathworks.com/help/deeplearning/ref/resnet18.html (Accessed: December 8).

[15] Torch Contributors, *ResNet*, Pytorch, 2024. Available at: https://pytorch.org/vision/main/models/resnet.html (Accessed: December 8 2024). 2, 3

[16] Yongdai Kim, Ilsang Ohn, and Dongha Kim, *Fast convergence rates of deep neural networks for classification*, Department of Statistics, Seoul National University, Seoul, Korea , 2019. Available at: https://machinelearningmastery.com/mcnemars-test-for-machine-learning/ (Accessed: December 8 2024). 5

[17] Torch Contributors, *Adam*, PyTorch, 2024. Available at: https://pytorch.org/docs/stable/generated/torch.optim.Adam.html (Accessed: December 8 2024). 2, 5

[18] E. Kavlakoglu and K. Kervin, *Classifying data using the SVM algorithm using Python*, IBM, 2024. Available at: https://developer.ibm.com/tutorials/awb-classifying-data-svm-algorithm-python/ (Accessed: December 8, 2024) 2

[19] Kurtis Pykes, *Cross-Entropy Loss Function in Machine Learning: Enhancing Model Accuracy*, datacamp, 2024. Available at: https://www.datacamp.com/tutorial/the-cross-entropy-loss-function-in-machine-learning (Accessed: December 8, 2024). 5

[20] Scikit-learn, *3.2. Tuning the hyper-parameters of an estimator*, Scikit-learn developers, 2024. Available at: https://scikit-learn.org/1.5/modules/grid_search.html (Accessed: December 8, 2024). 2, 5

[21] Keras, *Keras Documentation: Adam,*, Available at: https://keras.io/api/optimizers/adam/ (Accessed: December 8 2024). 4