**IMAGE CLASSIFICATION USING CNN AND SVM**

**Introduction:**

Image classification is an integral capability for modern computer vision systems. Recent breakthroughs in deep convolutional neural networks (CNNs) have led to immense progress in this domain. In this project, I tried to recreate an image classification model using CNN and SVM and aimed to know how SVM can use CNN’s framework to function better for Image classification and give more efficient results. The dataset I used for this project was taken from Kaggle called “Cards Image Dataset”. All images in the dataset have been cropped so that only the image of a single card is present, and the card occupies well over 50% of the pixels in the image. There are 7624 training images, 265 test images, and 265 validation images. The train, test, and validation directories are partitioned into 53 sub-directories, one for each of the 53 types of cards.

In the literature review, we explore highly relevant studies and methodologies on applying CNNs and SVM for image classification and compare how they perform based on the results. We will also look at how we can use SVM and CNN in coordination to make the model even more efficient and robust. Next, I provided a comprehensive background on the dataset and elucidated the CNN architecture chosen for this project. Then I also delineate the experimental framework, including model training, evaluation protocol, and performance metrics. In the results and analysis, we can see the presented quantitative outcomes as well as visualizations and examples to offer insight into the model.

**ABOUT THE DATASET:**

This dataset comprises high-resolution images of playing cards, each with dimensions 224 x 224 x 3 in jpg format. All images within the dataset have been meticulously cropped, ensuring that a single card dominates over 50% of the pixels. The dataset is divided into three sets: 7624 images for training, 265 for testing, and 265 for validation. The division is organized into 53 subdirectories, with each subdirectory representing one of the 53 unique types of cards.

**METHODOLOGY:**

**DATA PREPROCESSING:**

The data that was collected contained 3 folders which were uploaded into my Google Drive where I accessed them through the Colab notebook. The ‘Test’ folder contained 53 sub-folders with the card names as the folder names and with each folder having multiple images of the card in many unique ways. The ‘Test’ and ‘Val’ folders follow the same too.

A card with red hearts on it

Description automatically generated

Fig 1.0 An element in the dataset

**Model Creation and Training:**

The model used in CNN is the EfficientNet-B0 model as it is pre-trained on large-scale datasets (e.g., ImageNet). Transfer learning with a pre-trained EfficientNet model allows leveraging knowledge gained from one dataset to improve performance on another, potentially smaller dataset. The global pooling layer is modified to use adaptive average pooling to ensure a fixed-size output regardless of the input size. The original classifier is replaced with an identity function, effectively removing the final fully connected layer so that SVM integration is possible.

**The steps involved with CNN:**

* A custom CNN architecture was designed and implemented using the EfficientNet-B0 base model.
* The model is trained on a dataset of images from the test folder present in Google Drive.
* Training involves optimizing the model parameters using the Adam optimizer and cross-entropy loss.
* The CNN model is evaluated on a validation set, and the training process is monitored for performance metrics.
* The model was trained for 8 Epochs where we see that it peaked around the 5th epoch which is the best fit for this model.

Once the CNN model is trained, we then proceed to test the results by testing it with the images in the “test” folder which shows us the accuracy of the model and how the prediction works. We then proceed to SVM by using the features extracted in the penultimate layer of the CNN model.

**Steps in the SVM Model:**

The Support Vector Machine (SVM) is employed as a complementary model to the Convolutional Neural Network (CNN) for image classification in this project.

* Features are extracted from the penultimate layer of the pre-trained EfficientNet-B0 CNN model.
* These features serve as high-level representations of the input images.
* An SVM classifier is trained using the extracted features from the CNN. The SVM builds a decision boundary to discriminate between different classes based on the learned features.
* The SVM model takes advantage of the rich image representations learned by the CNN during pre-training.
* SVM operates on these representations to make predictions, allowing it to generalize well to unseen data.

A screenshot of a computer program

Description automatically generated

Fig 1.1 The Epoch training

A graph with blue and orange lines

Description automatically generated

Fig 1.3: The graph showing Training and Validation loss in the epochs.

**LITERATURE REVIEW:**

In the paper "Image Classification using SVM and CNN" which is the base of the project they compare the image classification techniques, starting with traditional machine learning using Support Vector Machines (SVM) and then moving to deep learning with Convolutional Neural Networks (CNN). They first implemented SVM on a small dataset of 350+ images over 5 classes, achieving 93% accuracy. But SVM performance saturates with more data. They augment the dataset to 3000+ images using techniques like color space changes, image translations/flips, etc. SVM accuracy reduces to 82% on this dataset. They then implement a CNN with architecture similar to LeNet-5. The CNN outperforms SVM with 93.57% accuracy on the same augmented dataset. The core components of the CNN are convolutional layers to detect features, max pooling to reduce spatial dimensions, flattening, and fully connected layers for classification. Comparative analysis shows deep learning techniques like CNN can continue improving with more data compared to traditional machine learning. This highlights the greater potential of deep learning for image classification. The methodology illustrates a structured approach to evaluating traditional ML vs deep learning techniques for image classification through rigorous experimentation. The CNN architecture and data augmentation strategies are also relevant techniques. Alternatively in a similar paper "An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image Classification", the paper explores using a Support Vector Machine (SVM) instead of a softmax classifier at the output layer of a Convolutional Neural Network (CNN) architecture for image classification. Typically, CNNs use a softmax classifier, but some studies have suggested SVM may perform better. They implement a simple CNN with 2 convolutional/pooling layers, connected to a SVM classifier. The CNN-SVM model is evaluated on the MNIST and Fashion-MNIST image datasets and compared to a CNN-Softmax model. On MNIST, CNN-Softmax achieves slightly better accuracy (99.23%) than CNN-SVM (99.04%). But on the more complex Fashion-MNIST dataset, their performance is comparable. The results do not confirm SVM being superior to softmax for CNN classification, but only a simple base CNN architecture was used. Further experiments with more complex CNN models and data preprocessing may better validate if using an SVM classifier can improve CNN performance for image classification. In conclusion, the paper experimentally explores the combination of CNN for feature extraction connected to an SVM for classification, suggesting promise but needing more research.

**RESULTS AND COMPARISONS:**

**CNN model:**

The CNN model, based on the modified EfficientNet-B0 architecture, was trained over 8 epochs on a dataset. The training and validation loss curves demonstrated a consistent decrease, indicating effective learning over the training period. This implies that the model was able to capture complex features in the images and generalize well to new, unseen examples.

A close-up of a card

Description automatically generated

Fig 1.4: Validating the Model

A close-up of a card

Description automatically generated

Fig 1.5: Another validation of the Model

**SVM Model:**

Features were extracted from the penultimate layer of the CNN model and used as input for training an SVM classifier. The CNN model's penultimate features served as rich representations of images, capturing high-level patterns learned during the training process. The SVM model achieved an accuracy of 100% on the validation set. This demonstrates the effectiveness of using extracted features from the CNN model for image classification using a traditional machine-learning approach.

**CONCLUSION:**

The integration of Convolutional Neural Networks (CNN) with Support Vector Machines (SVM) in the context of image classification presents a compelling approach that leverages the strengths of both deep learning and traditional machine learning methodologies. This integration was explored and implemented using a custom CNN model based on the EfficientNet-B0 architecture, with features extracted from the penultimate layer for training an SVM classifier. The following key conclusions can be drawn from this project:

* The CNN model, with its ability to capture hierarchical features, serves as a powerful feature extractor. Features extracted from the penultimate layer encapsulate high-level representations of images, providing rich information for subsequent classification using SVM.
* The SVM classifier, trained on the extracted CNN features, achieved 100% accuracy on the validation set. This highlights the effectiveness of utilizing the penultimate layer features as inputs to a traditional machine learning model for image classification.
* The integration of CNN with SVM provides a balance between the computational efficiency of SVM and the feature learning capabilities of CNN. The training times for the CNN and SVM models were [mention training times], showcasing the potential for combining the efficiency of SVM with the representational power of CNN features.
* The use of transfer learning with a pre-trained CNN model facilitated the extraction of meaningful features without the need for extensive training on the specific image classification task. This accelerates the development cycle and enhances model performance.
* The integrated approach demonstrated robust performance on the validation set. Further investigation is warranted to evaluate the generalization capabilities of the integrated model across diverse datasets and real-world scenarios. Future work could explore fine-tuning strategies and additional optimizations for improved performance.

In conclusion, the integration of CNN with SVM proves to be a synergistic strategy for image classification, capitalizing on the feature learning capabilities of CNN and the classification robustness of SVM. This project provides valuable insights into the potential of combining deep learning and traditional machine learning for enhanced image classification tasks.

**REFERENCES:**

* Agarap, A. F. (2019, February 7). An architecture combining convolutional neural network (CNN) and Support Vector Machine (SVM) for Image Classification. arXiv.org. <https://arxiv.org/abs/1712.03541>
* S. Y. Chaganti, I. Nanda, K. R. Pandi, T. G. N. R. S. N. Prudhvith and N. Kumar, "Image Classification using SVM and CNN," 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA), Gunupur, India, 2020, pp. 1-5, doi: <https://ieeexplore.ieee.org/abstract/document/9132851>.
* Dataset : <https://www.kaggle.com/datasets/gpiosenka/cards-image-datasetclassification/code?datasetId=2579480&searchQuery=pyto>