BUILDING ROBUST CLASSIFIERS FOR REAL-TIME DECISION SUPPORT WITH LOW-SHOT LEARNING

by

SANDIPANI BASU

(Under the Direction of Suchendra M. Bhandarkar)

ABSTRACT

Image classification is a computer vision task that involves categorizing images into certain predefined classes based on their visual features. In certain real-world scenarios where images are captured in varying conditions, real-time classification needs to be done under artificial or environmental stressors such as occlusion, camouflage, image distortions etc. The state-of-the-art image classification models require large amounts of data to build a robust and resilient classifier unaffected by stressors. We propose a supervised low-shot learning approach to improve image classification on a limited dataset in a stressed environment by incorporating shape-based feature representations along with the high-level CNN image features. In this thesis we show that the proposed fusion model improves upon the benchmark classification accuracy tested on a dataset of military vehicles under varying battlefield stressors. We visually represent the model performance using F1-scores and ROC-AUC plots as performance metrics.

INDEX WORDS: Image classification, Low-shot learning, shape context descriptor, feature fusion, Convolutional neural networks

# BUILDING ROBUST CLASSIFIERS FOR REAL-TIME DECISION SUPPORT WITH LOW-SHOT LEARNING

by

SANDIPANI BASU

BE, Sinhgad Academy of Engineering, India, 2016

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2023

# BUILDING ROBUST CLASSIFIERS FOR REAL-TIME DECISION SUPPORT WITH LOW-SHOT LEARNING

by

SANDIPANI BASU

| | |
|---|---|
| Major Professor: | Suchendra Bhandarkar |
| Committee: | Lakshmish Ramaswamy |
| | Khaled Rasheed |

Electronic Version Approved:

Ron Walcott
Vice Provost for Graduate Education and Dean of the Graduate School
The University of Georgia
August 2023

# DEDICATION

I am honored to have gotten this opportunity to purse my Masters research at the prestigious, University of Georgia. I would like to dedicate this thesis to my parents, brother and sister-in-law who have been a constant source of support and inspiration throughout my life. I would also gratefully mention the University of Georgia football team of 2021 who won the National Championship after 41 years and showed us the true meaning of perseverance and hard work, inspiring us all. Go Dawgs.

ACKNOWLEDGEMENTS

I would like to thank all my committee members for their time, patience, encouragement, and invaluable insights throughout my research. A special thanks to my major professor Dr. Bhandarkar and Patrick Debroux from US DEVCOM for their mentorship and their expertise that significantly improved the quality of the research. I would also like to thank all my friends here at University of Georgia who helped me keep moving and remain grounded. I am extremely grateful to the School of Computing and the University of Georgia for providing me with the resources needed to successfully complete my Masters Thesis.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

# CHAPTER 1

# INTRODUCTION

In recent years, artificial intelligence (AI) has had a significant influence on warfare. Its integration into military systems and operations has transformed various aspects of warfare such as building autonomous frameworks comprising of unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs), powering decision-support systems, improving target identification, tracking and engagement capabilities among others [25]. The state-of-the-art AI-powered decision-making systems analyze vast amounts of data from sensors and other sources, providing situational awareness and generating actionable intelligence [26]. In the domain of target identification, AI-enabled systems are being used to determine targets with increased accuracy by identifying discriminative features in a scene of multiple targets with complex backgrounds [27]. Similarly, surveillance, reconnaissance and combat vehicles performing autonomous operations without direct human control in real-time is an immediate result of machine intelligence [28]. Automatic target recognition (ATR) systems require enormous amounts of training data to reliably identify and localize the target in battlefield scenes under a variety of environmental and viewing conditions. On account of these battlefield conditions, ATR systems must be capable of recognizing targets that they have not been explicitly or adequately trained on. These include previously unseen, or rarely seen target classes or more importantly known targets corrupted with certain interference and limiting factors such as noise, occlusion, lighting, and illumination variations among others.

Convolutional Neural Networks (CNNs) and Deep Neural Networks (DNNs) have revolutionized the field of computer vision. They are playing a crucial role in empowering computers to achieve human-level classification performance on target recognition tasks. These neural network architectures excel in accurately classifying objects and patterns in images by utilizing hierarchical layers of interconnected artificial neurons to automatically learn and extract complex features from raw image data. In certain tasks,

depending on their depth and complexity, neural networks, they often rival closely or even surpass human performance in recognizing objects. However, data collection and data availability to train these networks are major hurdles in many domains of interest. For some, it might be challenging and expensive to acquire well-annotated data, while other areas might have restricted access to such data due to privacy or ethical concerns. There are strategies to deal with this drawback which involve transfer learning where pre-trained models are fine-tuned on smaller datasets, or data augmentation techniques which expands the dataset size by applying transformations on the image (rotation, translation, scaling, cropping etc.). Some problem domains have experienced an ever-increasing number of target categories without adequate observed samples of these targets. This often leads to an ineffective and inaccurate classification of previously unseen or seen targets under very stressed conditions. This bottleneck is addressed by Zero-Shot Learning (ZSL) and Low Shot Learning (LSL) approaches where ample training data per target category or class is not strictly required. ZSL is essentially designed for situations where the target is absent during training whereas in LSL the target is insufficiently represented in the training data. [15][17]

LSL or more commonly called few-shot learning (FSL) is a research area where final predictions must be made based on a few training samples. There is a subtle difference between the traditional supervised learning predominantly used and the ZSL/LSL methods. In addition to training and generalizing the model over the test set, the model, through its training phase also learns to differentiate or find similarities between the target objects. The LSL technique encourages the model to be more robust and invariant to environmental variations in visual features thus maximizing classification accuracy. ZSL on the other hand goes a step further, where the model learns how to classify novel target classes. For instance, a model trained to distinguish between a car and bike is given an image of a plane. The categories used in the training set are thus known as the “seen” samples while the unlabeled training instances are the “unseen” classes.

Humans perform ZSL and LSL naturally. Even though any single object can project an infinity of image configurations to the retina, it is only with rare exceptions that an image fails to be rapidly and readily classified, either as a familiar object category or as an instance that cannot be classified (itself a form of

classification) [1]. The object can be occluded by other objects, behind a foliage or even missing some parts or just a novel exemplar of a particular category, humans still do not fail to classify it [1]. To achieve this with machines, we would need vast amounts of training data of each category to handle such variations.

The proposed project aims to provide formal techniques and metrics to assess the accuracy and robustness of zero-shot and low-shot machine learning methods in the context of automated decision making in real-time scenarios. In this project we propose a novel and comprehensive framework to test the limits of LSL by inferring several environmentally stressed target categories from a few seen target categories. The project aims to build a model fusing deep features extracted from CNNs pre-trained on ImageNet weights and Shape Context descriptor as auxiliary information [2][3] to improve on the state-of-the-art CNNs for image classification. This proposed framework will help us determine the functional dependence of the classification accuracy on global features, extracted from the state-of-the-art CNNs and a geometric representation of the shape of an object in the form of the Shape Contexts descriptor. We expect the framework to aid in the training process yielding a higher classification accuracy when seen target classes are tested under battlefield stressors (*Distortion*, *Camouflage*, *Defilade*, *Vertical coloration*, and *Horizontal coloration*). The conceptual framework should also give us insights into the inference process in situations where the feature fusion aids in correctly identifying a target category and performing exceptionally well in comparison to the state-of-the-art and in situations where it is still bested by the CNNs.

# CHAPTER 2

# BACKGROUND AND LITERATURE REVIEW

## 2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have radically changed the entire field of computer vision, enabling pivotal advancements in various tasks such as object detection, segmentation, image classification, even autonomous vehicles and many more. The reason why CNNs gained immense popularity is their ability to learn hierarchical representations from raw image data, eliminating the need for feature engineering.

One of the pioneering CNN architectures was introduced by LeCun et. al (1998) [4], known as the LeNet-5. This model demonstrated the power of convolutional layers, pooling and fully connected layers specifically designed for handwritten digit recognition. This work laid the foundation for all the subsequent developments in CNNs.

The AlexNet proposed by Krizhevsky et al (2012) [5], is a deep CNN architecture that won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012. A deeper architecture with multiple convolutional and fully connected layers, dropout regularization and the use of Rectified Linear Units (ReLUs) [38] helped them significantly outperform other methods. AlexNet paved the way for several deep CNN architectures, namely VGGNet [6] with its primary focus being the depth of the network architecture. GoogLeNet [7] introduced inception modules as being computationally more efficient. It achieved competitive performance on ImageNet with fewer model parameters in comparison to its predecessors. This was pivotal for the development of more lightweight CNN architectures.

The ResNet (residual network) [8] was proposed to address the vanishing gradient problem which had plagued previous very deep networks. This architecture achieved state-of-the-art performance on several visual recognition tasks and became a fundamental building block for deeper architectures.

The use of CNNs was extended beyond image classification. The advent of Region-based CNNs (R-CNN) [9] combining CNN features and region proposal algorithms for object detection was game changing. This also led to the development of Faster-R-CNN [10] and Mask-R-CNN [11] which achieved state-of-the-art performance in object detection and instance segmentation.

Additionally, CNNs have been successfully applied to semantic segmentation tasks. Fully Convolutional Networks (FCN) [12] introduced the idea of replacing fully connected layers with convolutional layers for dense pixel-wise predictions. U-Net [13] proposed an encoder-decoder architecture with skip connections, improving the accuracy of segmentation networks, especially in biomedical imaging.

Recent advancements in CNNs include attention mechanisms [29], where models can focus on relevant image regions, and self-supervised learning, which leverages unsupervised pretraining to boost performance with limited labeled data.

In conclusion, CNNs have emerged as a dominant paradigm in computer vision, enabling significant breakthroughs in various tasks. The continuous development of new architectures, optimization techniques, and applications has led to remarkable progress in the field, and CNNs remain at the forefront of state-of-the-art solutions for image analysis related problems.

### 2.2. Zero shot and Low shot learning

**Definition 1.1.** Zero-Shot Learning (ZSL) is a Machine Learning paradigm where a pre-trained deep learning model is made to generalize on a novel category of samples, i.e., the training and testing set classes are disjoint. [43]

ZSL aims to recognize and classify objects or concepts for which no labeled examples are available during training. Instead, ZSL leverages auxiliary information, such as semantic attributes, textual descriptions, or class embeddings, to bridge the gap between seen and unseen classes.

The seminal work by Lampert et al. [14] introduced the concept of attribute based ZSL, where each class is associated with a set of semantic attributes. By learning the relationships between attributes and visual features, ZSL models can infer the presence or absence of attributes for unseen classes, enabling accurate classification. Later, novel approaches were proposed to improve ZSL performance. Akata et al. [15] introduce the use of Convolutional Neural Networks (CNNs) to map visual features to semantic embeddings, while Socher et al. [16] utilize recursive neural networks for compositional phrase embeddings.

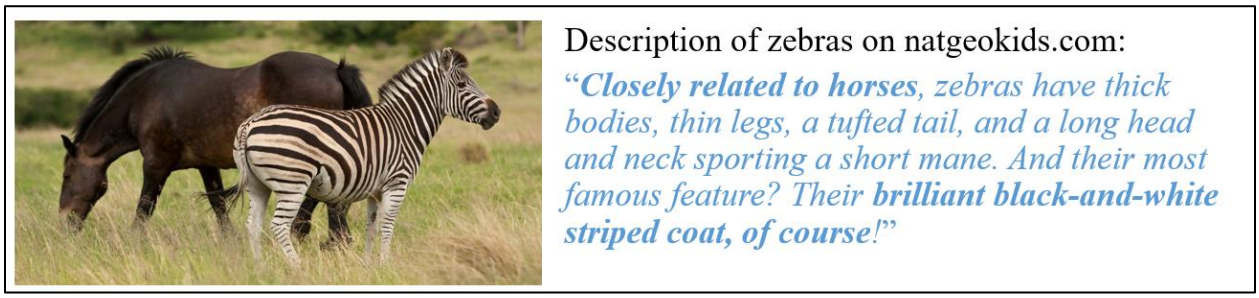



**Figure 1:** Zero-shot learning example. A model which has not been trained on zebra images (unseen sample) can still predict the said unseen sample accurately with auxiliary knowledge such as a textual description.

**Definition 1.2**. Low-Shot Learning (LSL) is a Machine Learning framework that enables a pre-trained model to generalize over new categories of data (that the pre-trained model has not seen during training) using only a few labeled samples per class. It falls under the paradigm of meta-learning (meta-learning means learning to learn). [42]

Low-shot Learning focuses on training models to recognize and classify novel classes with only a limited number of labeled examples per class. LSL tackles the challenge of generalizing to unseen classes by learning from few-shot examples and leveraging knowledge from seen classes.

Matching Networks [17] introduced the concept of learning a metric space where instances from the same class are closer than instances from different classes. Prototypical Networks [18] extended this idea by learning a prototype representation for each class based on the few available examples, enabling efficient classification of novel classes.

To further enhance LSL performance, meta-learning approaches such as Model-Agnostic Meta-Learning (MAML) [19] and Meta-Learning with Memory-Augmented Neural Networks (MANNs) have been introduced. These methods aim to learn an optimization procedure that adapts the model quickly to novel tasks with few-shot examples. Relation Networks [20] utilize deep neural networks to learn relationships between support examples and query examples, capturing the similarities and differences between classes for improved few-shot learning. Graph Neural Networks (GNNs) [30] have also been employed to model relationships and propagate information across instances.

Recent advancements have also explored the combination of ZSL and LSL techniques, termed Generalized Zero-shot Learning (GZSL) [31]. GZSL aims to bridge the gap between seen and unseen classes while addressing the challenges of limited labeled data, enabling more comprehensive and flexible learning scenarios.

In conclusion, ZSL and LSL have gained significant attention in the field of machine learning. These approaches tackle the challenges of recognizing and classifying unseen classes with limited labeled data, leveraging auxiliary information, generative models, metric learning, and meta-learning techniques. Continued research and development in these areas holds promise for expanding the capabilities of machine learning models in handling novel and limited data scenarios.

### 2.3 Shape context descriptor

The Shape Context (SC) descriptor, introduced by Belongie et al. [2][3], captures the distribution of local shape features around each point on a shape contour. The SC descriptor is widely used in shape matching, retrieval, and recognition tasks. A formal definition of the descriptor is given as

**Definition 1.3** (Shape context descriptor). Consider n points sampled on the shape contour. Consider the set of vectors originating from a sample point $p_i$ on the shape contour to the remainder n-1 contour sample points on the shape. For the point $p_i$, a coarse histogram $h_i$ representing the distribution of the relative positions of the remaining n-1 sample contour points are calculated and defined as the shape context of $p_i$. [2].

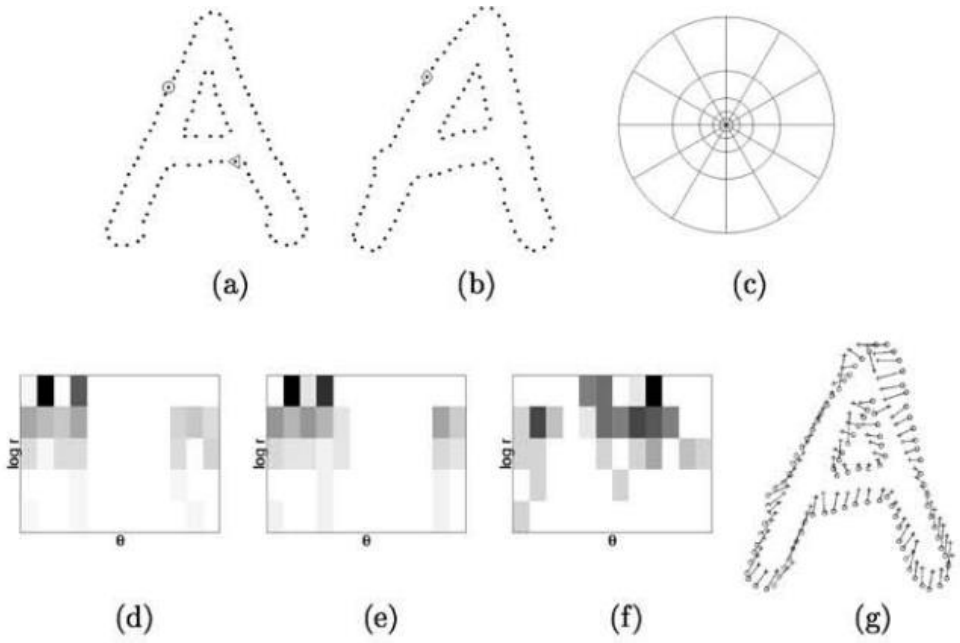



Figure 1: Shape context computation and matching. (a,b) Sampled edge points of two shapes. (c) Diagram of log-polar histogram bins used in computing the shape contexts. We use 5 bins for $\log r$ and 12 bins for $\theta$. (d-f) Example shape contexts for reference samples marked by $\circ, \diamond, \triangleleft$ in (a,b). Each shape context is a log-polar histogram of the coordinates of the rest of the point set measured using the reference point as the origin. (Dark=large value.) Note the visual similarity of the shape contexts for $\circ$ and $\diamond$, which were computed for relatively similar points on the two shapes. By contrast, the shape context for $\triangleleft$ is quite different. (g) Correspondences found using bipartite matching, with costs defined by the $\chi^2$ distance between histograms.

**Figure 2:** Example of shape context descriptor computation and matching as described in [2]

The key idea behind the SC descriptor is to represent each contour point by a histogram that encodes the relative spatial relationship between the reference sample point and the remaining sample points on the shape contour. The histogram bins represent different angular sectors and radial distances, thus capturing the local shape structure. By comparing the histograms of two shapes, similarity scores can be computed to determine the shape correspondence or match shapes in a database. The SC descriptor has shown robustness to shape variations, deformations, and occlusions [2]. Extensions of the SC descriptor have been proposed to handle rotation, scale, and translation invariance [3].

The Generalized Shape Context (GSC) descriptor, introduced in [21], extend the original SC descriptor by incorporating more contextual information. The GSC descriptor captures the relationships

between multiple points on the shape contour, enabling more comprehensive shape representation. In the GSC descriptor, each contour point is represented by a set of contextual descriptors, instead of a single histogram as in the SC descriptor. The GSC descriptor encodes pairwise relationships between the point of interest and other points on the contour, considering the distances, angles, and orientations. These pairwise descriptors provide a richer representation of the shape and capture more global shape information.

The GSC descriptor has demonstrated improved performance in shape matching and recognition tasks compared to the SC descriptor. The inclusion of contextual information helps handle occlusions, non-rigid deformations, and partial shape similarity. Extensions of the GSC descriptor have been proposed to handle scale, rotation, and affine transformations.

The applications of the SC and GSC descriptor span various domains, including object recognition, shape-based image retrieval, hand gesture recognition, and character recognition. Both descriptors have influenced subsequent developments in shape analysis and matching techniques, and they continue to be widely used and cited in the field of computer vision.

In conclusion, the SC descriptor and GSC descriptor have been instrumental in shape representation and matching tasks. These methods provide effective ways to capture and compare the local and contextual information of shapes, enabling robust and flexible shape analysis in various computer vision applications.

### 2.4 Multimodal feature fusion

Multimodal feature fusion has gained significant attention in recent years due to the increasing availability of data from various modalities, such as images, text, audio, and sensor data [39]. By leveraging complementary information from multiple modalities, multimodal feature fusion aims to enhance the representation power and performance of machine learning models in tasks such as classification, recognition, and retrieval. Early approaches in multimodal feature fusion focused on simple concatenation or early fusion, where features from different modalities are combined at the input level [39]. However, these methods often face challenges related to heterogeneity and different dimensionalities of modalities, which can limit the effectiveness of fusion.

Late fusion techniques [41] emerged as an alternative, where features from individual modalities are processed separately through individual models, and their predictions or representations are combined at a later stage. This approach allows for flexibility in modeling and can handle modalities with varying dimensionalities and complexities. Techniques such as decision-level fusion, score-level fusion, and feature-level fusion have been explored in late fusion [41]. Applications of multimodal feature fusion span various domains, including multimedia analysis, human-computer interaction, healthcare, and autonomous systems. It has been successfully applied in tasks such as multimodal sentiment analysis, audio-visual speech recognition, multimedia retrieval, and multimodal medical image analysis [41].

In conclusion, multimodal feature fusion plays a crucial role in leveraging the complementary nature of multiple modalities to enhance the performance of machine learning models. From early fusion to late fusion, deep learning-based approaches, and graph-based fusion methods, the field continues to evolve with the aim of better representing and utilizing multimodal data in various applications.

# CHAPTER 3

# TARGET CLASSES

As per the requirements of the US Army, five target classes of military vehicles were considered initially:

1. *Abrams* tank,
2. *Bradley* tracked armored fighting vehicle.
3. *HMMWV* high mobility multipurpose wheeled vehicle
4. *MRAP* mine-resistant ambush-protected light tactical vehicle
5. *Stryker* armored infantry personnel carrier.

Figure 3 shows the RGB images of the above target classes. An RGB image dataset was created with the images acquired from public domain sources accessed via the Internet. The dataset size was 26k images. An 80-10-10 split among training, validation and testing sets was used.


**Figure 3**: Sample military vehicle images obtained from the web.

Another dataset of 3D model renderings was used. These were obtained as 5-degree increments in the azimuth dimension, 90 and 60 degrees in the polar dimension [22]. We call this the *model dataset*, and it was used to train and test our SC descriptor framework [2].

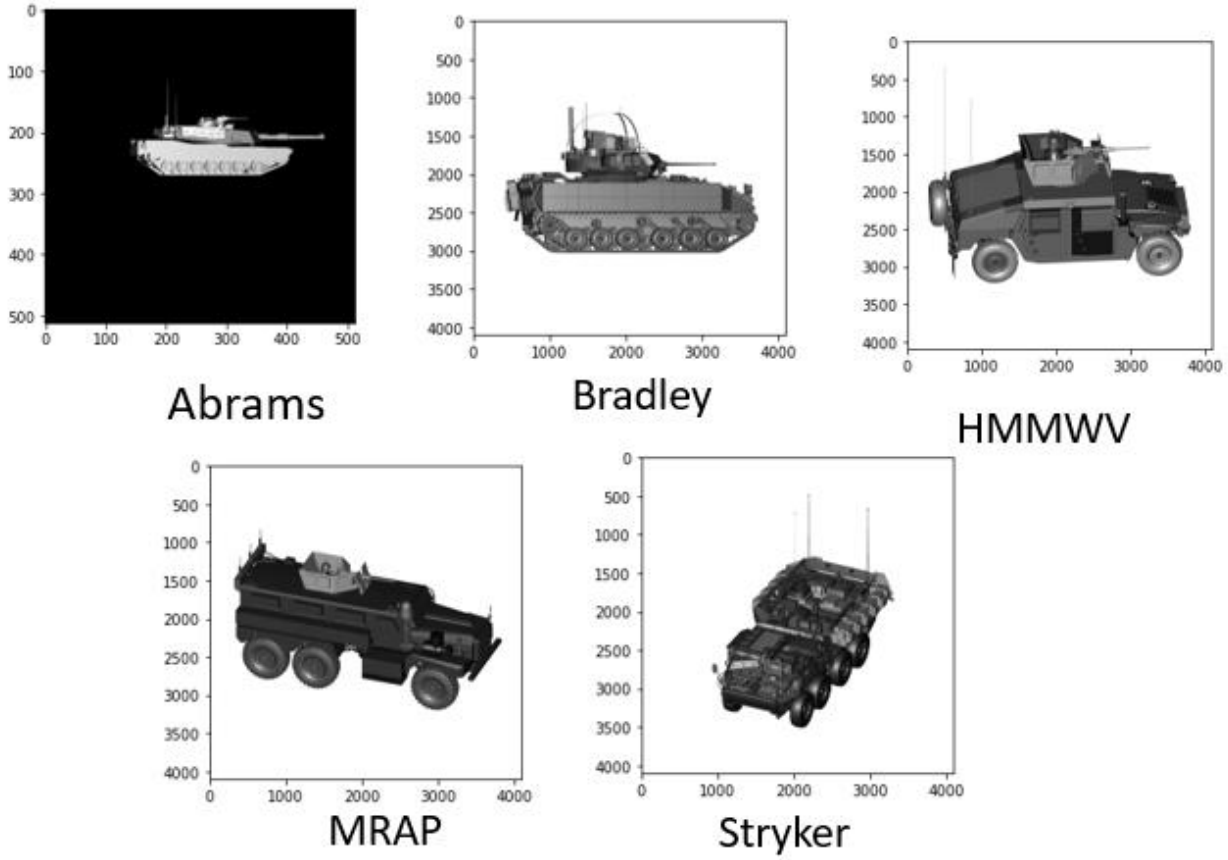



**Figure 4**: Model images of military vehicles

In this thesis, our primary aim is to test our proposal on a set of stressed images. Since our dataset is of military vehicles, we simulate common battlefield stressors on our images to test the robustness of our classification model. All the stressors are applied over the target military vehicle in 5% increments of the total target area [33]. A detailed description of each stressor type is as follows:

1. *Camouflage [33]:* The image target and the background are isolated using a K-means clustering based segmentation. Random background pixels were arranged on the target as patches. To simulate an increase in percentage of the *Camouflage*, the randomized background pixels were grown to cover a percentage of the target image.

2. *Coloration [33]:* This form of stressor disrupts object outlines. Variable-width stripes were superimposed on the target with incremental amounts of contrast. The stripes are both horizontal and vertical, so this gives us two different kinds of stressors.
3. *Distortion [33]:* This stressor was simulated by dilating the pixels of the target image. This dilation is done on random pixels on the target. To isolate the target, a similar approach as used in *Camouflage* was used.
4. *Hull-Defilade [33]:* To mimic the trenching of military vehicles, a similar method as *Camouflage* was used to generate masks and then patches were incrementally filled from the bottom of the mask, thus simulating an exposed un-trenched top of the vehicle.

# CHAPTER 4

# AUTOMATIC TARGET RECOGNITION METHODS

Our aim in this thesis is to compare different methods of automatic target recognition (ATR) and test the efficacy of these methods on a test-set of images that are corrupted with natural and human-induced stressors such as *camouflage, coloration, defilade, distortion*. Broadly, the proposed research compares the following methods:

1. Pure SC descriptor-based classification.
2. Pure CNN based classification.
3. CNN and SC descriptor fusion based classification.

The first section of this chapter primarily focusses on target class classification using solely shape information i.e., shape context descriptor [2]. The training is performed using 3D model renderings and a nearest neighbor approach is used to determine the closest target match. It is common knowledge that the Convolutional Neural Network (CNN) has been used extensively in computer vision tasks related to ATR such as image recognition, object instance detection/localization, object instance recognition and semantic segmentation. In section 2 we discuss image classification using CNNs namely Resnet50, InceptionV3 and VGG16. We define this type of target classification as *PureCNN*, and reference it as such throughout the rest of the document. Most of the examples in the open domain research literature that deal with computer vision applications of the CNN assume that sufficient training data is available and/or a CNN that is sufficiently pretrained on a general dataset such as MS COCO [23] or ImageNet [24] can be easily fine-tuned on a limited training set of images pertaining to a specialized domain via a process of transfer learning. However, this assumption is severely tested when testing sets of images are corrupted with natural and human-induced stressors such as *camouflage, coloration, defilade,* and *distortion*. Figure 6 shows

examples of such stressed images. Thus, in section 3 we introduce a novel approach to improve the robustness of CNN classifiers using late feature fusion [41]. We define this model as the *CoNNText* model and reference it as such throughout the document. SC descriptors are used as a source of auxiliary information. The final object classification is performed using the fusion of deep features extracted using the CNN and the shape information feature vectors derived using the SC descriptor.

### 4.1 Pure Shape Context Descriptor-based Classification

The SC descriptor [2] is a highly discriminative feature descriptor which is used to measure similarity between shapes for the purpose of object recognition. The measurement of similarity is preceded by first determining the corresponding contour points between two shapes under the assumption that a pair of points that is determined to be a true correspondence would have similar SC descriptor. That would mean, for a reference point p on the shape contour, the histogram of the relative coordinates of the remaining n-1 points will constitute the SC descriptor of that reference point [2]. For this purpose, the SC descriptor is attached to each contour point. The optimal set of corresponding points between the two shapes is determined using bipartite graph matching that is cast as an optimal assignment problem. Second, given the optimal point correspondences, a transformation that best aligns the two shapes is determined. Recognition is performed in a *k*-nearest-neighbor (*k*-NN) classification framework which determines the maximally represented class within the *k* most similar stored prototype shapes or exemplars to the query shape within the image.

We used model images of the 5 classes of military vehicles as our dataset to implement a purely SC-based classifier using the *k*-Nearest neighbor approach. We converted the model images (Figure 4) to silhouette data using a MATLAB script (Figure 5). Using the silhouette images (132 different views of each target class), we used 3, 5 and 7 *k*-NN classifiers to predict our target classes.

The SC descriptor algorithm used is from [2]. Below is a brief step by step algorithmic description of the implementation:

**Input**: Set of points representing a shape (e.g., contour points or key point locations)
**Output**: Shape context descriptor for each point in the shape

**Define parameters:**

Number of angular bins: *num_bins* ←*nb*
Number of radial bins: *num_rings* ← *nr*
Radius for computing shape context ← *r*
Reference point for computing angles: *ref_point*

**Compute Euclidean distance between each point and the reference point:**

Create an empty array to store distances for each point: *distances.*
For each point p in the shape:
Compute the Euclidean distance between p and *ref_point.*
Store the distance in the *distances* array.

**Compute angles between each point and the reference point:**

Create an empty array to store angles for each point: angles.
For each point p in the shape:
Compute the angle between the line connecting p and *ref_point* and the x-axis (or any desired reference direction)
Store the angle in the angles array.
Initialize an empty shape context descriptor for each point: *shape_contexts.*

**For each point p in the shape:**

Initialize an empty histogram for p in the shape context descriptor: *histogram.*

**For each neighboring point q around p within the specified radius:**

Compute the Euclidean distance between p and q ← $dist_{pq}$.

**Compute the angle between the line connecting p and q and the x-axis**: *angle_pq* ← $\theta_{pq}$.

**Compute the angular bin index based on the angle: bin_index** ← $angular_{index}$

$\lfloor(\theta_{pq}/(2 * pi) * nb)\rfloor$

**Compute the radial bin index based on the distance: ring_index** ← $radial_{index}$

$$\left\lfloor\left(\frac{\log\left(\frac{\text{dist}_{pq}}{r}\right)}{\log\left(\frac{\text{dist}_{max}}{r}\right)}\cdot nr\right)\right\rfloor$$

**Increment the corresponding bin in the histogram**:

*hist* [$angular_{index}$] [$radial_{index}$] += 1

**Normalize the histogram values:**

Calculate the sum of all bin values in the histogram:
*hist_sum* = sum(*hist*)

**Normalize each bin value by dividing it by *hist_sum*.**

**Store the normalized histogram in the shape context descriptor for p**:

*shape_contexts*[p] = *histogram*

**Return the shape context descriptor shape_contexts for each point in the shape.**

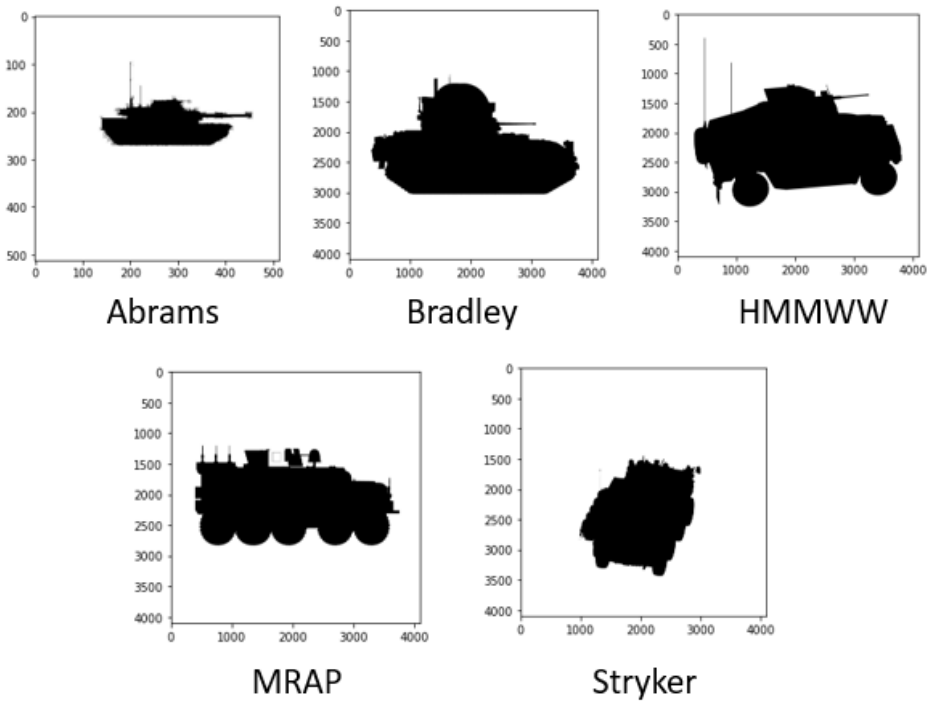



**Figure 5**: Silhouettes extracted from model images of military vehicles.

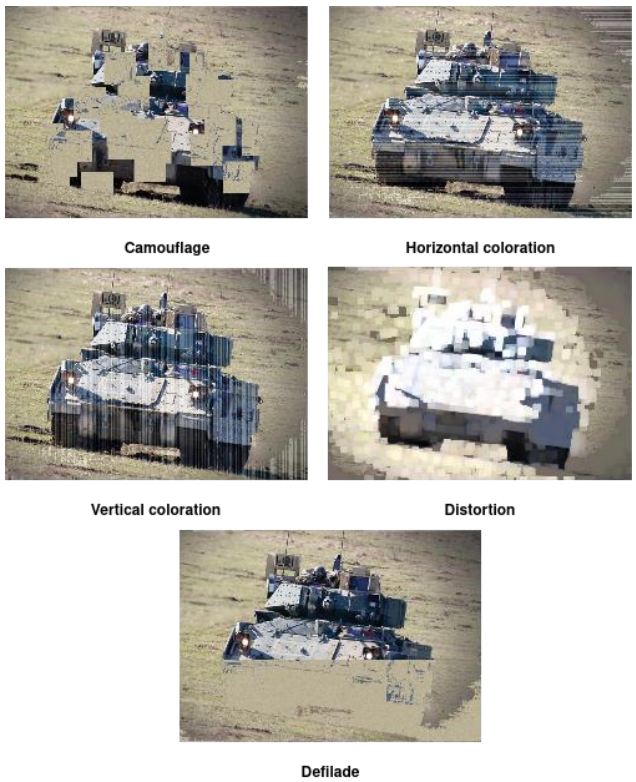



**Figure 6:** Images of Bradley at 50% stress level for all stressors.

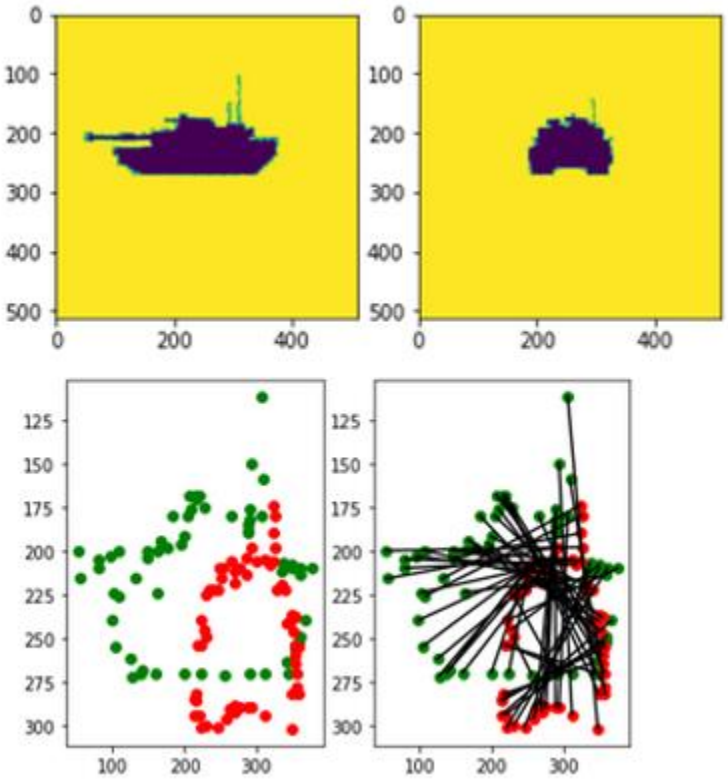



**Figure 7**: Visual representation of shape matching.

### 4.2 Pure CNN-based Classification.

In this thesis the *PureCNN* as defined earlier is a CNN-based image classifier. We employ a state-of-the-art CNN pretrained on ImageNet [24] that is specifically designed to classify images into pre-defined classes. The training dataset used in this research are 5 classes of military vehicles namely *Abrams, Bradley, MRAP, HMMWV* and *Stryker*. The size of the entire dataset is 28k and employ a 80-10-10 split for training, validation and testing respectively. Given the size of our training set and unique classification domain we use transfer learning to fine tune the pre-trained models. Finetuning a CNN model involves taking the pretrained model and continuing the training process on a smaller data set, that is usually specific to a particular task or domain. During finetuning, the weights of the pretrained model are updated to better fit

the task or domain-specific data set. With transfer learning [36] [37] we can achieve significantly higher performance on a new task than training from scratch on a new task using a small training dataset. We use three well-known image classification models: Resnet-50, Inception-V3 and VGG-16. In all of these models, we do not include the final classification head and add our own classification layer. This is an important step during transfer learning to help the model learn task and target specific features.

**VGG16**

The VGG16 CNN architecture was developed by the Visual Geometry Group (VGG) [6]. As the name suggests, it consists of 16 layers. The convolutional layers have small 3 * 3 filters, capturing the local patterns in the image followed by ReLU [38]. It also includes a max pooling layer [38] with 2 *2 pooling windows and a stride of 2 for reducing the spatial dimensions. The VGG16 is a rather simple deep learning model with the last 3 layers being fully connected layers. We should note that due to its architecture, VGG16 is computationally expensive and may require significant resources to train from scratch.

**InceptionV3**

The InceptionV3 [34] was introduced by Google in 2015 and is 42 layers deep. The primary building block of this model is the inception module [34]. These modules are built to capture and extract visual features at various scales. It is a multi-branch structure consisting of 1x1,3x3 and 5x5 convolutional filters concatenated with max pooling layers [34]. One of the major modifications to InceptionV3 as compared to its predecessors [7] is use of factorizing convolutions [34]. This essentially meant replacing a 5x5 convolutional layer by two 3x3 convolutions thus reducing the total number of parameters thereby also reducing the computational cost. These features offer a good balance between good accuracy and computational efficiency.

**Resnet50**

The ResNet50 architecture is a CNN model that is 50 layers deep [8]. The Resnet50 model is divided into multiple stages, each containing multiple residual blocks [8]. The first stage consists of a single convolutional layer, followed by a max pooling layer. The output of the first stage is then passed through subsequent stages, each containing several residual blocks. A residual block in the Resnet50 [8] architecture

consists of multiple convolutional layers, followed by a shortcut connection that skips the convolutions and adds the input to the output of the block. This creates a residual connection that allows for the gradient to flow more easily and prevents vanishing gradients. Each residual block consists of three convolutional layers [8]. The first convolutional layer has a 1x1 kernel size, followed by a second convolutional layer with a 3x3 kernel size. The final convolutional layer has a 1x1 kernel size and doubles the number of filters. The shortcut connection is added to the output of the final convolutional layer, and the entire block is passed through a ReLU [38] activation function. It also includes global average pooling and a fully connected layer to produce the final output. Global average pooling [38] computes the average value of each feature map, resulting in a single feature vector for each channel. The fully connected layer then produces the final classification output.

### 4.3 CNN and Shape Context Descriptor Fusion-based Classification.

A new model termed *CoNNText* is proposed for ATR based on the fusion of the shape context (SC) descriptor with CNN-derived color and texture features to enhance the robustness of target classification. The combination of the shape and structural information provided by the shape context descriptor with the deep features provided by the CNN, is expected to result in more robust classification especially in images wherein the objects are occluded and/or subject to natural and human-induced stressors. Thus, *CoNNText* is a combination of *PureCNN and SC descriptor*.

Extracting the shape context vector for a query target in an RGB image poses several challenges. First, the object of interest needs to be segmented in the input image, i.e., the foreground object needs to be extracted from the background followed by extraction of the object contour using Canny Edge detector. Second, the shape context descriptor is computed from the extracted contour. It is essential to generate accurate segmentation masks to compute the shape context descriptor accurately and avoid the introduction of noise in the SC descriptor computation. As shown in Figure 8, straightforward contour extraction methods [2] when applied to real-world RGB images results in missing true contour pixels and the introduction of noisy contour pixels. This could be attributed to various factors such as poor illumination, specular reflection, poor contrast with background, *camouflage*, etc. The missing and noisy contour input

adversely affects the quality of the shape context descriptor extracted for the target object in the test image. Hence it is critical to improve the shape context information extracted from the RGB images.

**Figure 8**: Missing contour pixels

**Mask generation**

Generation of the shape context descriptor for an object needs to be preceded by the computation of an accurate object instance mask in the image which essentially captures the object silhouette. The initial attempts at accurate object instance mask generation were based on foreground-background discrimination using a CNN architecture where the CNN was trained to extract foreground object segmentation masks using a data set of training images. The first attempt used the YOLO-v3 [32] object detector model to generate the bounding box for the foreground object in the image where the YOLO-v3 [32] object detector was pretrained on the ImageNet weights. The shape mask was extracted by performing erosion, dilation, and thresholding on the bounding box contents. While the object detection was observed to be only partially accurate in terms of the bounding box, there were instances where certain parts of the target vehicle were missing in the bounding box (Figure 9).

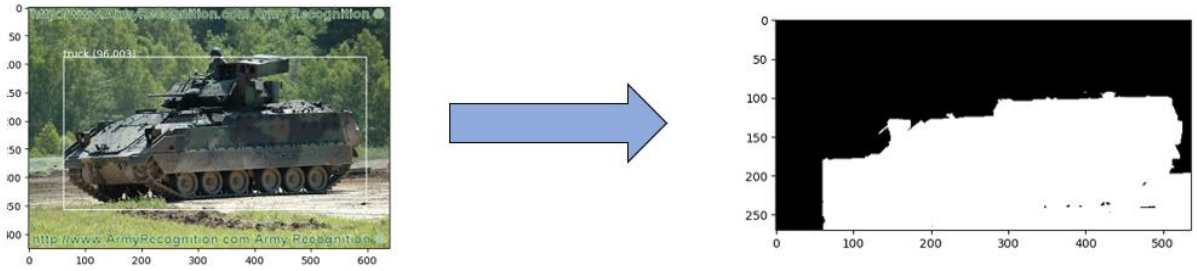



**Figure 9:** YOLO-v3 object detection and segmentation resulting in inaccurate contours

To improve target mask prediction a Mask-Region-based-CNN (M-RCNN) model [11] based on a ResNet-101 CNN backbone was employed. The M-RCNN model was pretrained on the MS COCO dataset [23] and finetuned on training images from the RGB dataset via transfer learning. The VGG Image Annotator tool was used to manually annotate 80 military vehicles to finetune the weights of the pretrained M-RCNN model. The training data set for the M-RCNN resulting from manual annotation is depicted in Figure 10. Figure 11 shows the initial results of the M-RCNN on a few target images. While the results captured the basic target shape silhouette, it missed finer details such as the turrets and antennae.


**Figure 10**: Manual annotations generated using VGG Image Annotator tool.

The initial results of the M-RCNN suggested that (a) high-quality model image data needs to be included in the training data set, and (b) higher-quantity and higher-quality manually annotated data is needed for training the M-RCNN. Consequently, the size of the RGB image training data set for the M-RCNN was increased and restricted to 100 high-quality RGB images. The criteria for selecting *high quality images* are as follows:

- Target objects should be free of occlusion.
- There should not be any secondary objects such as humans, trees, and other military vehicles in the image.
- The target object should not be trenched or camouflaged.

The M-RCNN was trained with these 100 manually annotated RGB images and the model images generated for creating the shape context exemplar library for each target class with the objective of training the M-RCNN with as clean training data as possible. The M-RCNN was first trained with the 100 RGB images for 30 epochs and the resulting weights saved. These trained weights were further refined by retraining the M-RCNN for 50 epochs with the model image data for each target class. The resulting images in Figure 12 show substantially improved target mask generation that captured more of the object details.

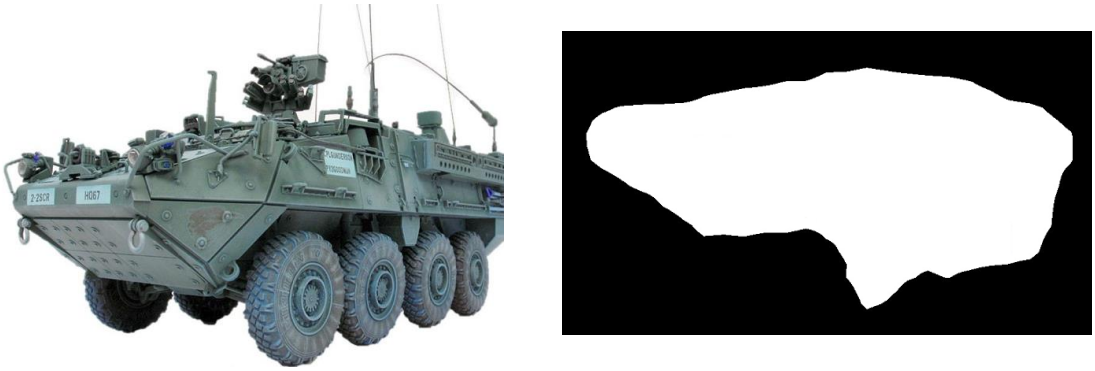


**Figure 11**:Mask generation missing out on finer details of the Stryker vehicle.

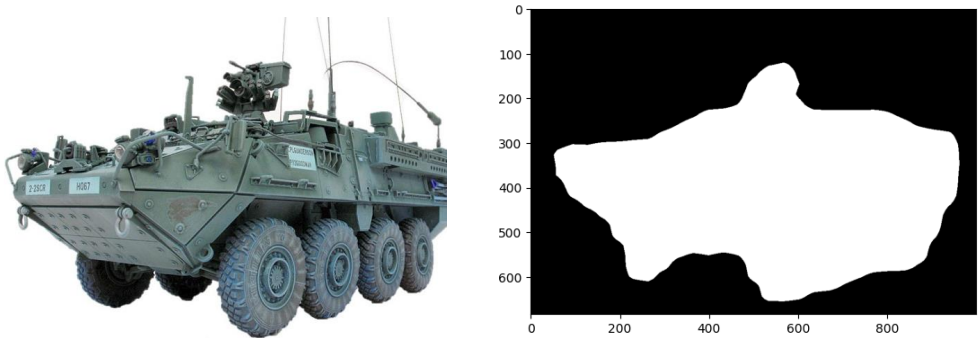



**Figure 12:** Improved masks after inclusion of model data in the training set.

**Feature Fusion**

The fusion of the deep image features and the shape context vectors are done at an early stage of the processing pipeline. This is called an early-stage fusion. Early-stage fusion has the advantage of preserving the raw information from multiple modalities and allowing the model to jointly learn from

different sources of input. It can enable the model to capture both low-level details and high-level correlations between said modalities. This means the modalities are fused before any significant high-level analysis or decision making is done. In this research the features to be fused are both visual features. The fusion of CNN image features and shape context vectors, early-stage fusion involves concatenating or merging the feature vectors obtained from CNNs and shape context descriptor before feeding them into subsequent layers or classifiers. This allows the combined information from both modalities to be processed and analyzed together from the beginning. The schematic of the proposed fusion is depicted in Figure 13.

The major challenge in doing so is the heterogeneity of the features. In our research the dimension of the extracted deep CNN features is (512) and that of SC descriptor is (6000,). The techniques to handle said difference involve techniques such as dimensionality reduction, feature selection or feature mapping to align the dimensions appropriately. We address this challenge by using an autoencoder [35]. Our goal is solely dimensionality reduction, so the network is designed to extract the most informative features and discard all the unnecessary and redundant information. The encoders consist of multiple layers, typically fully connected or convolutional layers. We use the ReLU [38] activation function to introduce non-linearity and allow the network to learn complex relationships between the features. The pivotal bottleneck layer in this architecture has a fewer number of neurons than the input layer forcing the network to learn a compact representation of the shape features. While building this network we focus on preventing overfitting, hence, we apply regularization techniques like dropout layers [38], L1 regularization [38] and Batch normalization [38].

Once we reduce the dimensionality, we add a set of fully connected layers or more commonly called a dense layer stack. The purpose of such a stack is to introduce nonlinear transformations, thereby learning complex relationships between the fused features. Essentially, we are learning higher level representations of the fused features to capture the abstract and discriminative features to aid in making an accurate prediction. The dense layer stack is followed by an activation function, ReLU [38] and finally the output layer, which is a softmax [38] layer, producing the final predictions based on the learned representations.

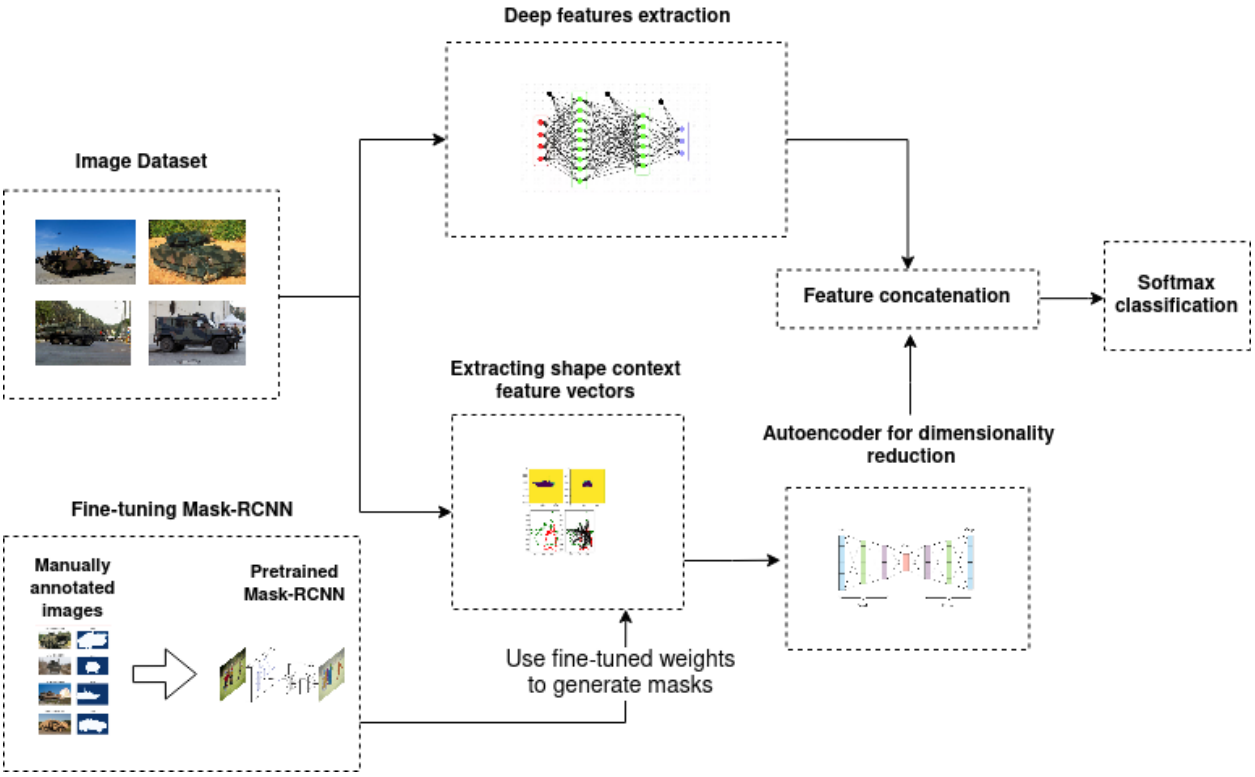



**Figure 13**: Proposed model architecture for feature fusion

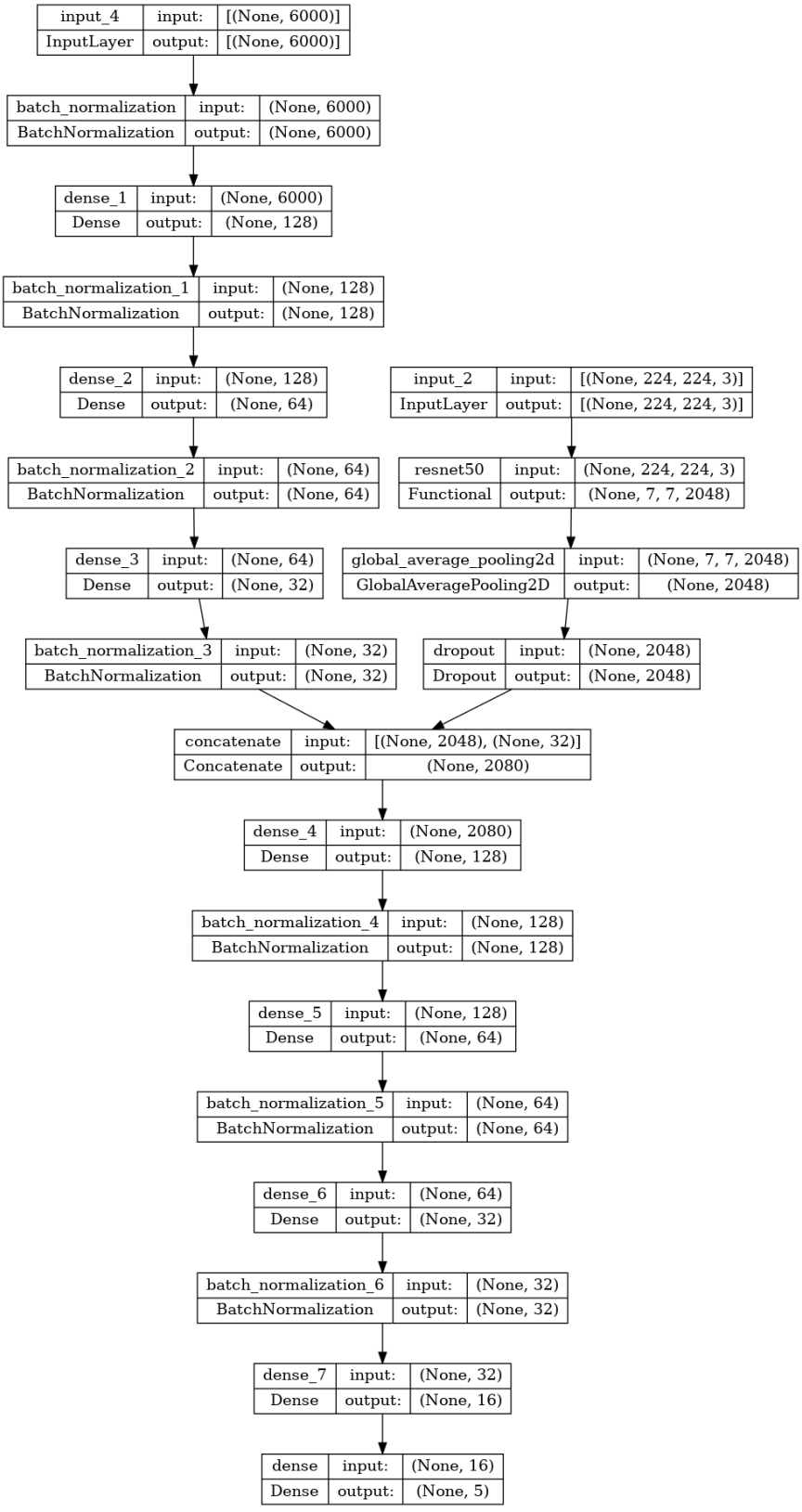



**Figure 14**: Model layers of the fusion model

The proposed model layers as outlined in Figure 14 has multiple moving parts which are explained in detail below.

- The first layer on the right branch is the Input layer which takes in a raw input image. This layer has no trainable parameters, its sole purpose is to pass the data onto the subsequent layers without any computation.
- The left branch has an input which accepts the shape context descriptor. The subsequent layers are a dense stack to learn the shape context features and reduce overfitting, hence the use of batch normalization [38]. We see that the features are reduced before concatenation.
- The Lambda layers on the right branch after the Input layer are the preprocessing layers which are a part of our pre-trained CNN model provided by the TensorFlow API. The preprocessing involves cropping the image to a dimension of 224x224x3. Other transformations include normalizing and scaling the pixel values.
- The functional CNN layers are the transfer learning part of our training.
- The global average pool layers have been added to reduce the spatial dimensions of a feature map. Unlike fully connected layers this dimensionality reduction does not depend on the input spatial dimensions. The reduction from a spatial feature map to a one-dimensional vector is performed by taking an average of each channel. This vastly reduces the number of parameters and prevents overfitting. An added advantage to our use case is that it makes the model more robust to translations to the input image by being insensitive to the exact spatial location of the features.
- The dropout layer is used to prevent overfitting in a deep learning model. When a model is too complex, it tends to fit the training data too closely but performs poorly on unseen data. It randomly sets some of the output values of a layer to zero during training. This means that the network cannot rely on any single input or feature during training and must learn to rely on a more robust combination of features instead. The dropout layer acts as a form of regularization by introducing noise and reducing dependencies between neurons. As the random units are dropped out, the model

is forced to learn more robust and generalized features that are not biased on specific input activations, hence improving the model's ability to generalize unseen data.

- The concatenate layer is used to combine outputs of multiple models or layers along a specified axis. Usually, it is the feature axis which is what we have used. This layer concatenates the feature vector and the shape context vector.
- We then add the classification head which is a dense stack of layers to learn concatenated features minimizing classification loss. Our experiments showed that the deeper the dense stack the better the results.
- The final layer is the dense prediction layer in this model. The number of output neurons is equal to the number of prediction classes.

# CHAPTER 5

# EXPERIMENTAL RESULTS AND DISCUSSION

## 5.1 Results of *k*-NN classification.

The evaluation of performance of the shape context descriptor in isolation was first considered. Of the 132 model silhouette images for each target class, 100 images were used to create the prototype or exemplar shape context descriptor for the class whereas the remainder 32 were used as the query or test images. The results of the *k*-NN classifier for $k = 3$, 5 and 7 are captured in the confusion matrices in Figure 15. For $k = 3$, the overall classification accuracy (averaged over all target classes) was observed to be 69.69% whereas for $k = 5$ and $k = 7$, the corresponding classification accuracy values were 71.96% and 72.72%, respectively. Figure 17 shows the nearest prototype silhouette image retrieved for a given query silhouette image for each target class using the shape context descriptor. As can be seen, the query silhouette image and the prototype silhouette image are visually very similar.

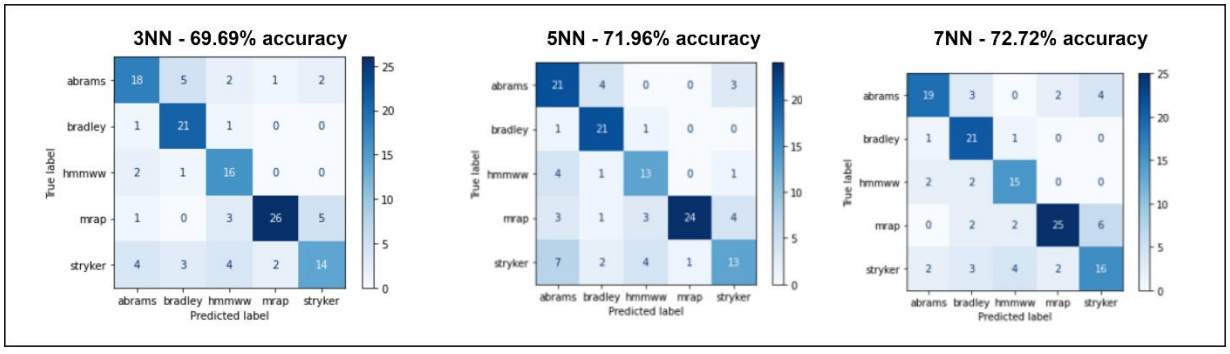



**Figure 15**: Pure shape context results

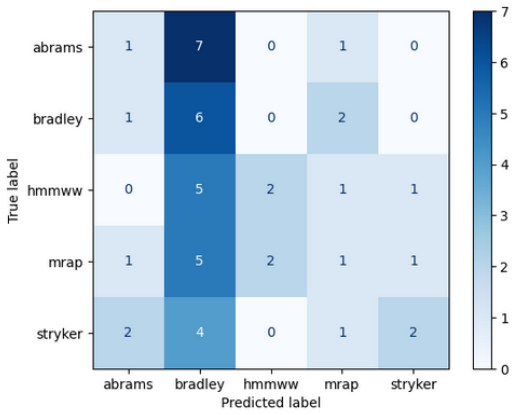



**Figure 16**: Confusion matrix for shape based RGB image classification.

Figure 16 is the confusion matrix of RGB test images tested on the trained *k*-NN classifier. Essentially, we perform this experiment to test the robustness of a naïve shape-based classifier on real-world image data. Also, note that the shape context descriptor of the RGB images are extracted from the masked silhouettes (as discussed in Chapter 4.3). The results clearly indicate that *the object shape* obtained from a real-world image in and of itself is not the best features for classification.

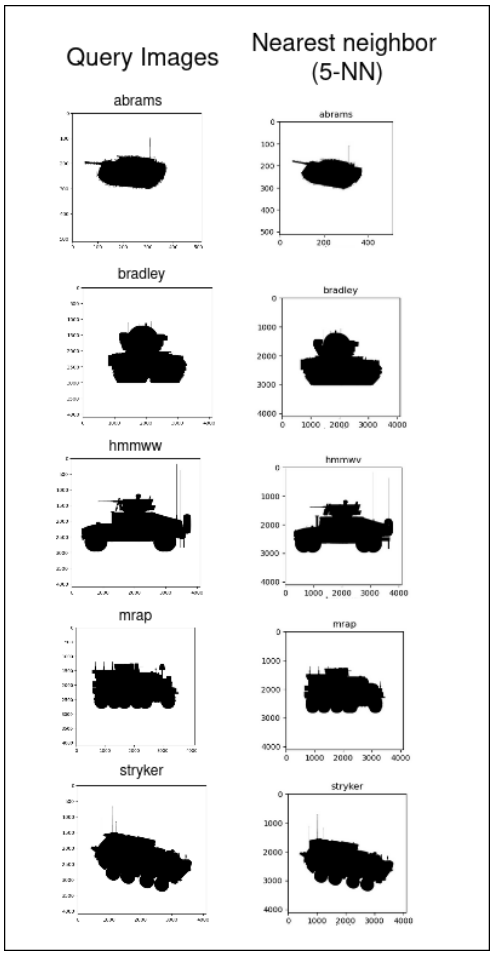



**Figure 17**: Nearest neighbors for a query shape from each target class.

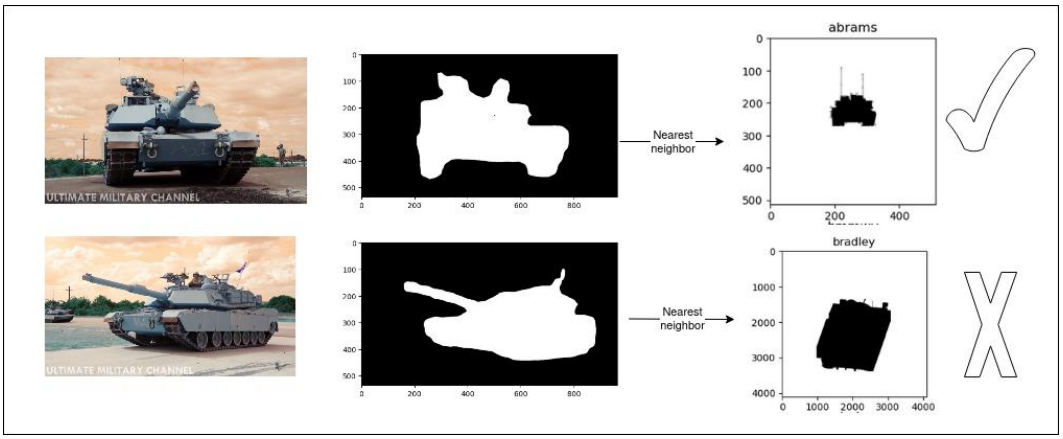



**Figure 18**: Pure shape context results on RGB data

### 5.2 Comparison of *PureCNN* and *CoNNText* classification.

We perform a holistic comparative analysis of the effect of *no stress, low stress, medium stress, and high stress* on the *PureCNN* and *CoNNText* models. The stress levels chosen are 0%, 25%, 50% and 75% respectively. In this section, we will first see how the fusion of shape information distinctly helps the overall model classification. Overall, we see the *accuracy* of both the *PureCNN* model and *CoNNText* model and compare them for each stressor level. *Accuracy* represents the proportion of correctly classified samples out of the total number of samples in the dataset, expressed as a percentage. We also compute the F1-scores and ROC-AUC values across stressors and target classes. The plots are represented as a bar chart with each row of the bar chart representing a target class and each column, a stressor. Lastly, we will look at how each of the three *CoNNText* models perform in comparison to each other.

Before we consider a more detailed analysis based on *F1-scores* and *ROC-AUC scores* here are a few initial observations based on Figures 19, 20 and 21 as follows:

1. The *CoNNText* Resnet50 clearly outperforms its *PureCNN* counterpart. The fusion of the shape information helps it achieve a better accuracy across all the stressors and through all the stress levels.
2. The *CoNNText InceptionV*3 does not do worse than the *PureCNN* exceeding the *PureCNN* accuracy levels only for *Horizontal coloration.*
3. The *CoNNText* VGG16 however is not able to capture the shape features effectively to boost accuracy. We will discuss this further in this chapter.

**Table 1:** Classification Accuracy of Resnet50 *PureCNN* vs *CoNNText*

| | | Stress Levels | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0% | | 25% | | 50% | | 75% | |
| | **Model Type** | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* |
| **Stress Type** | Distortion | 97.00% | 97.90% | 92.00% | 90.62% | 64.99% | 75.00% | 50.00% | 51.00% |
| | Camouflage | 98.00% | 98.95% | 79.00% | 85.41% | 57.99% | 67.70% | 40.00% | 44.79% |
| | Vertical Coloration | 98.00% | 98.95% | 89.99% | 97.90% | 86.00% | 84.38% | 66.00% | 66.66% |
| | Horizontal Coloration | 98.00% | 98.95% | 93.00% | 94.79% | 62.99% | 86.45% | 36.00% | 64.58% |
| | Defilade | 98.00% | 97.91% | 93.99% | 94.38% | 69.99% | 81.25% | 52.99% | 62.50% |

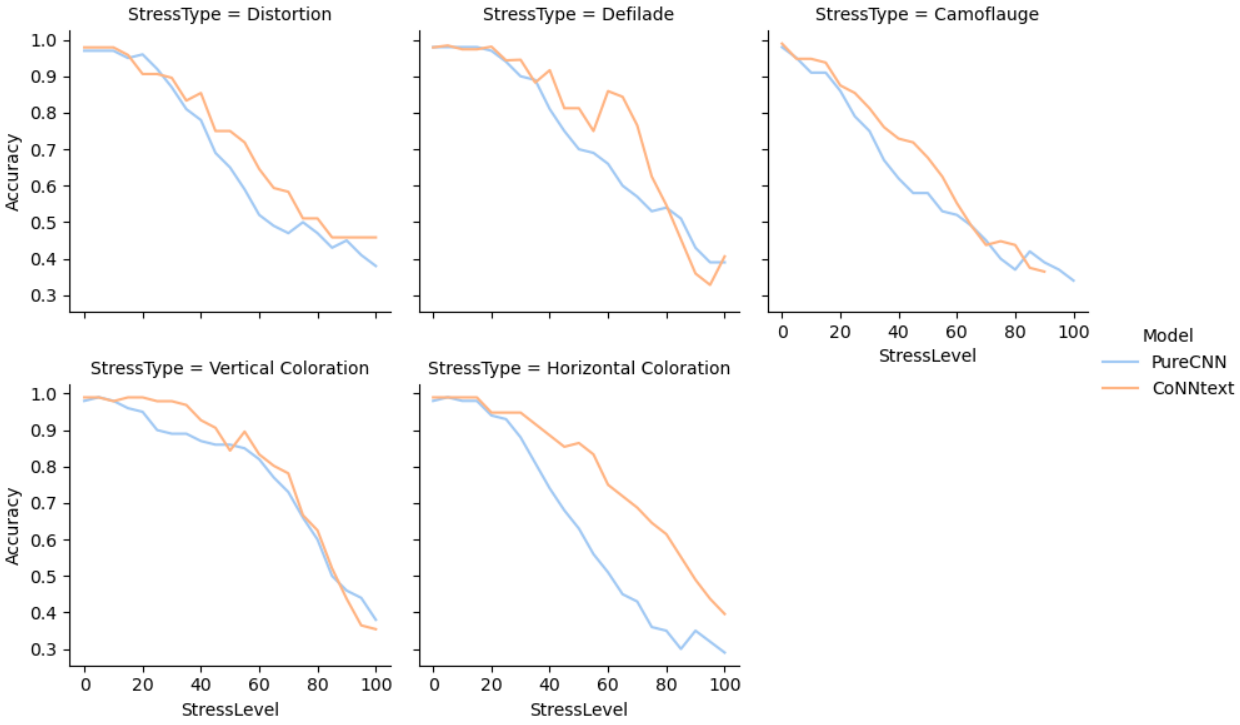



**Figure 19**: Stress-wise accuracy plots for Resnet50

**Table 2:** Classification Accuracy of InceptionV3 *PureCNN* vs *CoNNText*

| | | Stress Levels | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0% | | 25% | | 50% | | 75% | |
| Stress Type | Model Type | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* |
| | Distortion | 99.00% | 98.95% | 98.00% | 93.75% | 81.99% | 72.91% | 47.99% | 50.00% |
| | Camouflage | 99.00% | 98.95% | 89.00% | 93.75% | 67.00% | 54.16% | 41.99% | 34.38% |
| | Vertical Coloration | 99.00% | 98.95% | 98.00% | 98.95% | 91.00% | 86.45% | 56.99% | 64.58% |
| | Horizontal Coloration | 99.00% | 98.95% | 93.00% | 96.88% | 75.00% | 80.20% | 40.99% | 61.45% |
| | Defilade | 99.00% | 98.95% | 95.99% | 90.63% | 70.99% | 80.20% | 55.00% | 46.88% |

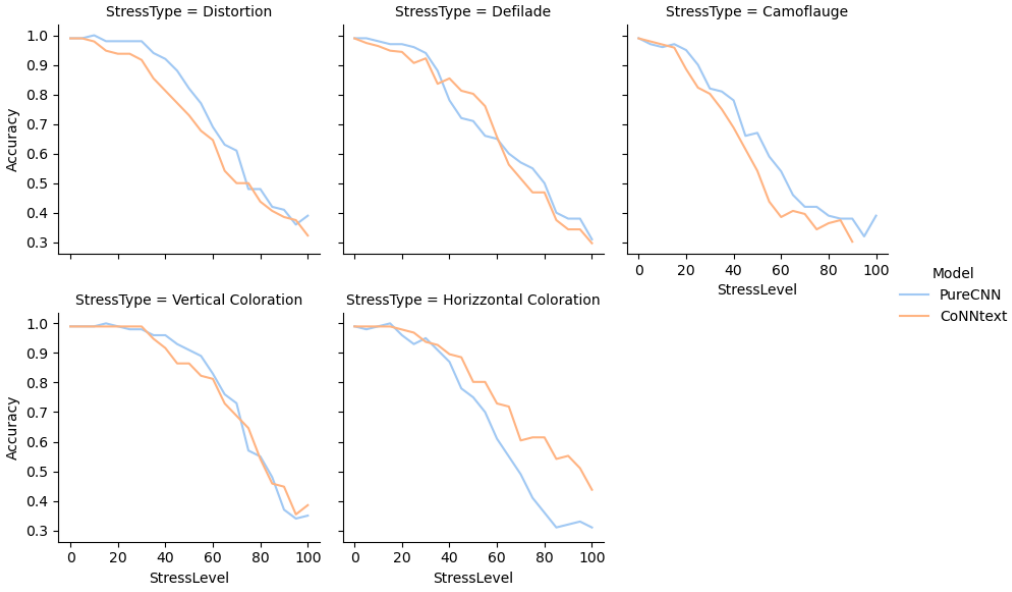



**Figure 20**: Stress-wise accuracy plots for InceptionV3

**Table 3:** Classification Accuracy of VGG16 *PureCNN* vs *CoNNText*

| | | Stress Levels | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | 0% | | 25% | | 50% | | 75% | |
| | Model Type | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* | *PureCNN* | *CoNNText* |
| **Stress Type** | Distortion | 88.99% | 84.38% | 76.99% | 70.83% | 47.99% | 52.10% | 34.99% | 35.41% |
| | Camouflage | 89.00% | 82.30% | 72.00% | 62.50% | 52.99% | 56.25% | 40.99% | 38.54% |
| | Vertical Coloration | 89.00% | 80.20% | 85.00% | 73.96% | 64.99% | 48.96% | 40.00% | 41.66% |
| | Horizontal Coloration | 89.99% | 80.20% | 77.00% | 75.00% | 43.00% | 53.13% | 40.00% | 33.33% |
| | Defilade | 92.00% | 82.29% | 83.99% | 68.12% | 64.99% | 44.79% | 43.00% | 48.43% |

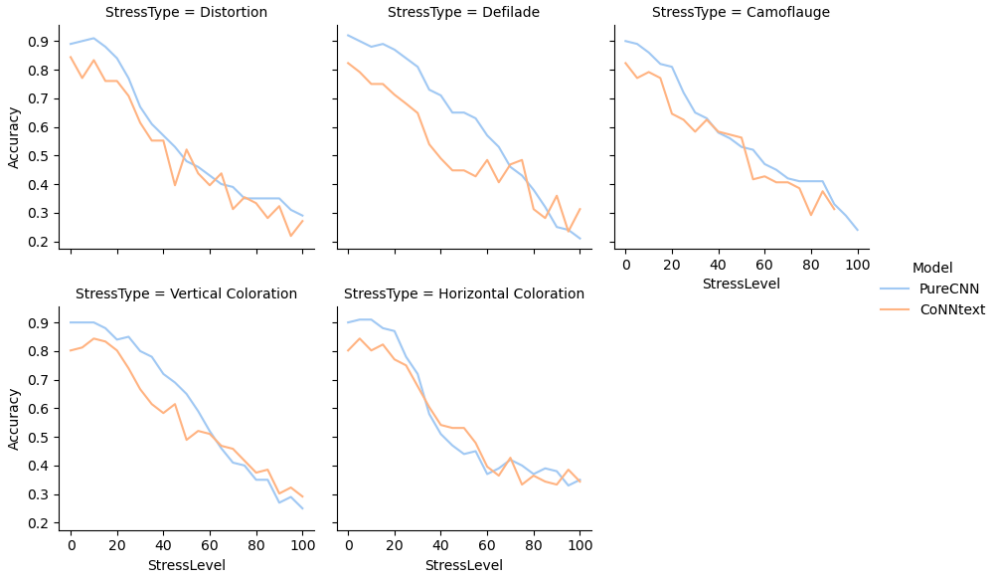



**Figure 21**: Stress-wise accuracy plots for VGG16


**Figure 22**: F1-score comparison (Resnet50 base). Top row to bottom row: target classes: *Abrams*, *Bradley*, *HMMWW*, *MRAP*, *Stryker*. Left column to right column: stressors: *Defilade*, *Distortion*, *Vertical Coloration*, *Horizontal Coloration*, *Camouflage.* X-axis: *Stress levels (0%, 25%, 50%, 75%).* Y-axis: *F1-score.* Blue bar – *PureCNN*, Orange bar – *CoNNText*

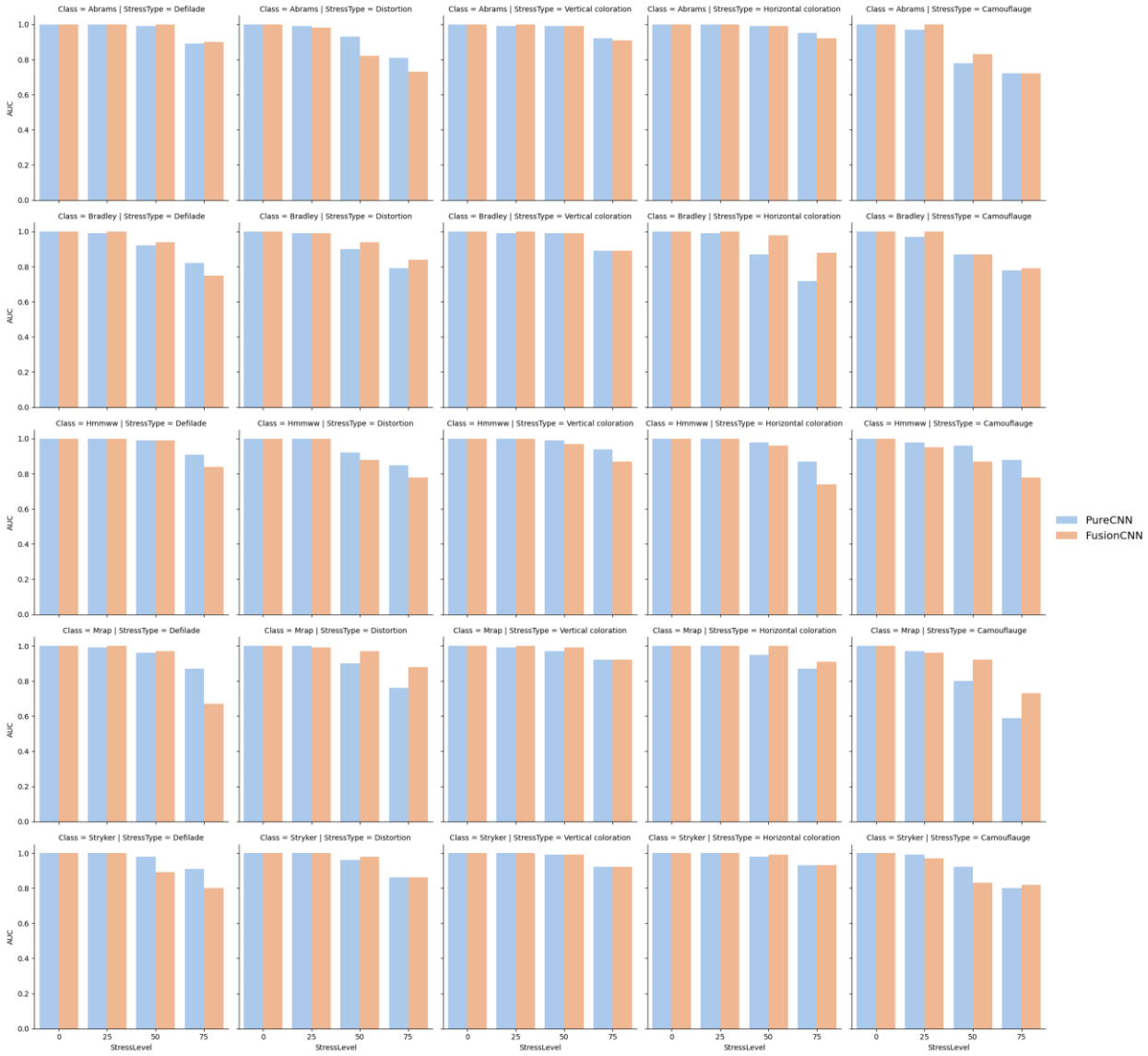


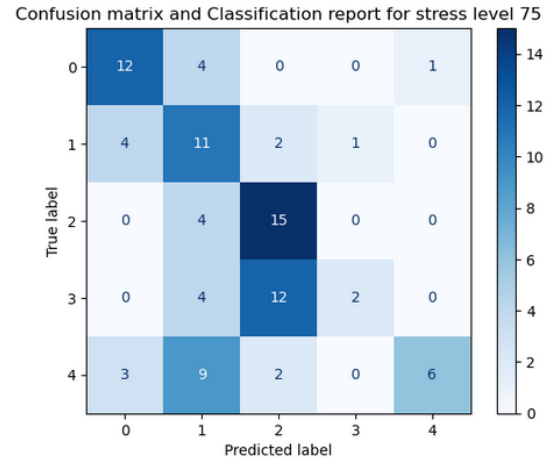

**Figure 23**: AUC score comparison (Resnet50 base): Top row to bottom row: target classes: *Abrams, Bradley, HMMWW, MRAP, Stryker.* Left column to right column: stressors: *Defilade, Distortion, Vertical Coloration, Horizontal Coloration, Camouflage.* X-axis: *Stress levels (0%, 25%, 50%, 75%).* Y-axis: *AUC-score.* Blue bar – *PureCNN,* Orange bar – *CoNNText*

**Detailed Stress-wise analysis (row-wise analysis of the bar charts)**

***Defilade***

**Observation 1:** At 25% and 50% stress levels, *CoNNText* performs better than the state of the art. *MRAP* shows considerable improvement in *CoNNText* in comparison to *PureCNN* owing to its unique shape and structure. We will see throughout our observations that *MRAP* and *Bradley* show consistently better performance over the *PureCNN*.

**Observation 2:** An important observation at 75% stress level, is the drop in accuracy for the *MRAP* vehicle class. *HMMWW* and *MRAP* both have boxy shapes. The model which relies on the shape information hence predicts some instances of *MRAP* as *HMMWW*. As we see from the confusion matrix in Figure 24, the precision is good, but recall tells us a different story. There is a distinct difference in shape of the box-like structure; *HMMWW* have more straight edges and flat surfaces whereas *MRAPs* are more angular and curved. Since this is a *Defilade* stress test, most object parts are blocked so we would need additional auxiliary information to deal with this kind of stress.




| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.63 | 0.71 | 0.67 | 17 |
| 1 | 0.34 | 0.61 | 0.44 | 18 |
| 2 | 0.48 | 0.79 | 0.60 | 19 |
| 3 | 0.67 | 0.11 | 0.19 | 18 |
| 4 | 0.86 | 0.30 | 0.44 | 20 |
| accuracy | | | 0.50 | 92 |
| macro avg | 0.60 | 0.50 | 0.47 | 92 |
| weighted avg | 0.60 | 0.50 | 0.47 | 92 |

**Figure 24**: Confusion matrix- 75% *defilade* stress.

***Distortion***

**Observation 1:** At 25% we see all classes perform almost the same as *PureCNN*, with *MRAP* and *Bradley* performing better while the rest slightly lower with a 0.02 - 0.04 % difference in accuracy.

**Observation 2: A**t 50% and 75% stress level, *Abrams* has a lower accuracy for the fusion model of 0.08% - 0.10%. If we look at the confusion matrix, we observe a high precision on the classification but a low recall. The reason being that *Bradley* has a much more prominent shape, with a larger turret size (to fit a crew of at least three), and an elevated position of that turret that is mounted higher on the hull. The silhouette of the *Bradley* is much more distinct. *Abrams* has a similar shape to *Bradley* but with a smaller turret and comparatively less prominent shape. Because of *Distortion*, the rounded contours of *Bradley* are slightly biased in our shape model resulting in some *Abrams* being misclassified as *Bradley*.

***Coloration***

**Observation 1:** The coloration stressors are where the *CoNNText* model performs better or the same as the *PureCNN*. It solves a major problem we observe in our *PureCNN* of a bias towards one of the classes (precisely *Abrams*) and it solves the low precision high recall problem effectively as we observe in Figure 25 in the *Horizontal coloration* confusion matrix at stress level 75% for both *PureCNN* and *CoNNText*.

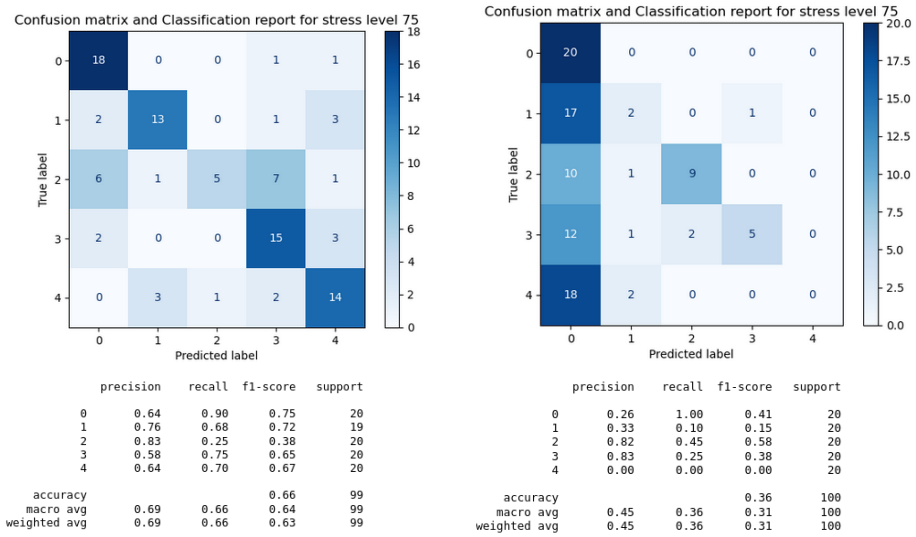



**Figure 25**: Comparison of confusion matrices CoNNText(L) vs PureCNN(R)

**Observation 2:** The reason we see a high difference in accuracy at 75% stress level for *Horizontal coloration* between *Abrams* and *Bradley*. The classification of *Abrams* and *Bradley* depends more on target outline than other vehicles [33]. Thus, in the presence of high *coloration* stress the shape of the target outline aids in the target recognition effectively. In the presence of the shape features the drawback of *PureCNN* architecture is overcome considerably.

***Camouflage***

**Observation 1:** *Camouflage* stress responds to the shape features very well. It shows better results across all stress levels and classes (except *HMMWV*) which we will discuss further when we examine the results per class.

**Detailed Class wise analysis (COLUMN-WISE analysis)**

***Abrams***

**Observation 1:** At a stress level of 25% the *Abrams* tank is correctly classified with our fusion model better than the state-of-the-art CNNs for most stressors reaching close to 99% accuracy for most except *Distortion*. As we see, it's *Distortion* where *Abrams* really suffers in *CoNNText*. A possible explanation could be that *Abrams, Bradley*, and *Stryker* are the three vehicles in our target classes which typically have guns or turrets mounted on top. So, as we see from the confusion matrix as well, the incorrect class assigned to *Abrams* for *Distortion* at 25% is mainly *Bradley*. Now the difference between these two vehicles is the length of the gun, the *Bradley* has a larger gun in comparison. *Defilade* and *Camouflage* is where it suffers again but only at a higher stress level i.e., 75% stress level.

***Bradley***

**Observation 1:** *Bradley* shows good classification results across the stress types except for *Defilade* at a 75% stress level, which is predominantly when the turrets are blocked since that is the most distinguishing feature for that tank.

***HMMWW***

**Observation 1:** *HMMWW* does not show much improvement with the addition of the shape features. An astute observation would be that *HMMWW* has no distinguishing shape features. Some instances of *HMMWW* have guns at top, but since it is a 'Multipurpose' vehicle, it goes through some structural variations hence making it inherently distinct and difficult to generalize based on shape features.

***MRAP***

**Observation 1:** *MRAP* shows the most promise with our model, due to its unique structure. *MRAP*s have the most unique shape with a high ground clearance and V-shaped hull at the back which makes its shape quite evident.

***Stryker***

**Observation 1:** *Stryker* are a class of tanks which show good results at 50% stress on a *Horizontal coloration*. A good explanation for that would be like *Abrams* and *Bradley*, the turrets are the most distinguishing features of the *Stryker*. In *Horizontal coloration*, the shape context descriptor capture well, the shape of a turret even in the presence of stress. So, all *Abrams*, *Bradley* and *Stryker* exhibit better performance for *Horizontal coloration* stress.


**Figure 26:** F1-score comparison of *CoNNText* models with different base CNNs. Top row to bottom row: target classes: *Abrams, Bradley, HMMWW, MRAP, Stryker.* Left column to right column: stressors: *Defilade, Distortion, Vertical Coloration, Horizontal Coloration, Camouflage.* X-axis: *Stress levels (0%, 25%, 50%, 75%).* Y-axis: *F1-score.* Blue bar – *InceptionV3,* Orange bar – *Resnet50,* Green Bar – *VGG16*

**Comparison of *CoNNText* with different base CNNs (Figure 26)**

1. We can clearly see that Inceptionv3 and ResNet50 perform at a similar level across classes and stressors. The VGG16 shows a significant drop in performance as compared to the other 2 models. There are a few reasons to this behavior:
    a. VGG was introduced with improvements to AlexNet in the form of multiple (3x3) kernel size filters in addition to the large filters. The configuration being used in this research is VGG16, which has 13 convolutional layers and 3 fully connected layers. This architecture has a series of small filters and max-pooling layers it results in a significant loss in spatial efficiency. The repeated stacking of the filters and the max-pooling layers leads to a reduction in spatial size, thereby leading to a loss of fine-grained or localized feature information, which in turn is important when testing on stressed images.
    b. The concatenated SC descriptor features and extracted deep features may contain redundant or conflicting information. In this case a shorter dense stack in Resnet50 effectively filters out noisy or redundant information resulting in better performance. Longer dense stack in VGG16 will be susceptible to overfitting or interference with the noisy and redundant features.
2. The Resnet50 on the other hand uses the residual or skip connections which enables information from earlier layers directly propagating to the deeper layers [8]. Specifically, the skip connections bypass one or more convolutional layers, allowing the gradient to flow from earlier layers to deep layers. This allows for direct information flow between different layers and enables important fine-grained features to directly reach the deeper parts of the network, skipping the down-sampling or max-pooling layers thus enabling localization effectively.
3. The Inceptionv3 on the other hand uses multi-scale convolutional filters i.e., a combination of 1x1,3x3 and 5x5 filters in parallel, thus helping the model to learn at various scales and maintaining a better spatial resolution.

# CHAPTER 6

# CONCLUSIONS AND FUTURE WORK

The results clearly show us that adding auxiliary information helps improve the accuracy of automatic target recognition in the state-of-the-art CNNs. In this research we have used shape information, in the form of SC descriptor in conjunction with deep features that are extracted by the *PureCNN* architectures. Note that we exploit both feature representations i.e., deep extracted features and shape features. When used together we leverage the local and global information underlying the shape and appearance of a target.

We presented a novel framework for fusion of CNN feature vectors and shape context descriptor for the automatic target recognition of military vehicles under environmental stressors. The proposed framework offers an alternative way of improving automatic target recognition for tasks which require a robust classifier that is immune to environmental noise and distortions. Such a robust classifier would aid a human in real-time decision making. The approach yields improved accuracy when tested on a dataset of stressed images from our target classes: *Abrams, Bradley, HMMWW, MRAP and Stryker*. The improvements in accuracy from +10% - +12% on a 50% *Distortion, Camouflage, Horizontal Coloration* and *Defilade* stressors and +12% improvement on a 75% *Defilade* stressed image. We see a significant increase in accuracy ranging from 36% to 64% on a 75% *Horizontal coloration* stressed image in our *CoNNText* Resnet50 model.

These results show us that low shot learning with auxiliary information on a domain specific task can help us in target recognition under environmentally stressed conditions. This research also paves the way for further research using additional information like textual data and domain specific 3-D point cloud information to further aid the proposed robust classifier. We successfully compared three state-of-the-art

CNN architectures and their corresponding *fusion* counterparts. The effect of feature fusion differs based on the baseline CNN architecture. We observe that the feature fusion works best in the case of Resnet50 and least in the case of VGG16. With InceptionV3 and Resnet50 *CoNNText* models performing comparatively well, we note that deeper architectures are more robust to target classes under stress, thereby also capturing more abstract features and higher-level semantics which complement the noisier local features derived from the shape of an object and give us a better overall classification result.

The current results on feature fusion using shape features are based on silhouettes extracted from a pretrained and fine-tuned Mask-R CNN model. Potential future work would be to improve the shape descriptor, by focusing more on contour extraction rather than segmentation. Apart from using shape information we can also use attribute information of the target classes, textual data on military vehicles and 3-D point cloud estimates as our sources of auxiliary information in the future.

## REFERENCES

[1] Biederman, I. (1987). Recognition-by-components: A theory of human image understanding. Psychological Review, 94(2), 115–147

[2] S. Belongie, J. Malik and J. Puzicha, "Shape matching and object recognition using shape contexts," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 4, pp. 509-522, April 2002, doi: 10.1109/34.993558.

[3] Belongie and Malik, "Matching with shape contexts," 2000 Proceedings Workshop on Content-based Access of Image and Video Libraries, Hilton Head, SC, USA, 2000, pp. 20-26, doi: 10.1109/IVL.2000.853834.

[4] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[5] Krizhevsky, A., Sutskever, I., & Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks (AlexNet) ImageNet Classification with Deep Convolutional Neural Networks (AlexNet) ImageNet Classification with Deep Convolutional Neural Networks.

[6] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[7] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[8] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[9] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2015). Region-based convolutional networks for accurate object detection and segmentation. IEEE transactions on pattern analysis and machine intelligence, 38(1), 142-158.

[10] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28.

[11] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

[12] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3431-3440).

[13] Falk, T., Mai, D., Bensch, R., Çiçek, Ö., Abdulkadir, A., Marrakchi, Y., ... & Ronneberger, O. (2019). U-Net: deep learning for cell counting, detection, and morphometry. Nature methods, 16(1), 67-70.

[14] Lampert, C. H., Nickisch, H., & Harmeling, S. (2013). Attribute-based classification for zero-shot learning of object categories. IEEE Transactions on Pattern Analysis & Machine Intelligence, (01), 1-1.

[15] Xian, Y., Lampert, C. H., Schiele, B., & Akata, Z. (2018). Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. IEEE transactions on pattern analysis and machine intelligence, 41(9), 2251-2265.

[16] Socher, R., Ganjoo, M., Manning, C. D., & Ng, A. (2013). Zero-shot learning through cross-modal transfer. Advances in neural information processing systems, 26.

[17] Vinyals, O., Blundell, C., Lillicrap, T., & Wierstra, D. (2016). Matching networks for one shot learning. Advances in neural information processing systems, 29.

[18] Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. Advances in neural information processing systems, 30.

[19] Finn, C., Abbeel, P., & Levine, S. (2017, July). Model-agnostic meta-learning for fast adaptation of deep networks. In International conference on machine learning (pp. 1126-1135). PMLR.

[20] Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., & Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1199-1208).

[21] Mori, G., Belongie, S., & Malik, J. (2005). Efficient shape matching using shape contexts. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(11), 1832-1837.

[22] Debroux, P. S. (2022). Analysis Methodology of Image Classifiers in Stressed Environments.

[23] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13 (pp. 740-755). Springer International Publishing.

[24] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). Ieee.

[25] J.R. Wilson (2019). Artificial intelligence (AI) in unmanned vehicles, https://www.militaryaerospace.com/

[26] U.S. Army CCDC Army Research Laboratory Public Affairs (2020), Army researchers augment combat vehicles with AI, https://www.army.mil/

[27] C. Wang, X. Liu, J. Pei, Y. Huang, Y. Zhang and J. Yang, "Multiview Attention CNN-LSTM Network for SAR Automatic Target Recognition," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 14, pp. 12504-12513, 2021, doi: 10.1109/JSTARS.2021.3130582.

[28] M. Zhang, J. An, D. H. Yu, L. D. Yang, L. Wu and X. Q. Lu, "Convolutional Neural Network With Attention Mechanism for SAR Automatic Target Recognition," in IEEE Geoscience and Remote Sensing Letters, vol. 19, pp. 1-5, 2022, Art no. 4004205, doi: 10.1109/LGRS.2020.3031593.

[29] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov,R. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in 32nd International Conference on Machine Learning (ICML 2015), vol. 37. PMLR, 2015, pp. 2048–2057.

[30] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner and G. Monfardini, "The Graph Neural Network Model," in IEEE Transactions on Neural Networks, vol. 20, no. 1, pp. 61-80, Jan. 2009, doi: 10.1109/TNN.2008.2005605.

[31] Chao, Wei-Lun, et al. "An empirical study and analysis of generalized zero-shot learning for object recognition in the wild." Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14. Springer International Publishing, 2016.

[32] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[33] Debroux, Patrick S. "Analysis Methodology of Image Classifiers in Stressed Environments." (2022)

[34] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826).

[35] Bourlard, H., & Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. Biological cybernetics, 59(4-5), 291-294.

[36] Perkins, D. N., & Salomon, G. (1992). Transfer of learning. International Encyclopedia of education, 2, 6452-6457.

[37] Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10), 1345-1359.

[38] Deep Learning (Ian J. Goodfellow, Yoshua Bengio and Aaron Courville), MIT Press, 2016.

[39] Jing Gao, Peng Li, Zhikui Chen, Jianing Zhang; A Survey on Deep Learning for Multimodal Data Fusion. Neural Comput 2020; 32 (5): 829–864. doi: https://doi.org/10.1162/neco_a_01273

jennifer.l.forsythe2.civ@army.mil

[40] Huang, S. C., Pareek, A., Seyyedi, S., Banerjee, I., & Lungren, M. P. (2020). Fusion of medical imaging and electronic health records using deep learning: a systematic review and implementation guidelines. NPJ digital medicine, 3(1), 136.

[41] Ramachandram, D. & Taylor, G. W. Deep Multimodal Learning: A Survey on Recent Advances and Trends. IEEE Signal Process. Mag. 34, 96–108 (2017).

The VGG16 model can achieve a test accuracy of 92.7% in ImageNet, a dataset containing more than 14 million training images across 1000 object classes.

[42] Everything you need to know about Few-Shot Learning https://blog.paperspace.com/few-shot-learning/

[43] The Essential Guide to Zero-Shot Learning [2023] https://www.v7labs.com/blog/zero-shot-learning-guide