Spring 2022

PORIFERAL VISION: Deep Transfer Learning-based Sponge Spicules Identification & Taxonomic Classification

Sudhin Domala
San Jose State University

PORIFERAL VISION: Deep Transfer Learning-based Sponge Spicules Identification & Taxonomic Classification

A project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements for the

Degree Master of Science

by

Sudhin Domala

May 2022

Advisor: Philip Heller

Committee Member: Nada Attar

Committee Member Amanda Kahn (Moss Landing Marine Laboratories)

ABSTRACT

The phylum Porifera includes the aquatic organisms known as sponges. Sponges are classified into four classes: Calcarea, Hexactinellida, Demospongiae, and Homoscleromorpha. Within Demospongiae and Hexactinellida, sponges' skeletons are needle-like spicules made of silica. With a wide variety of shapes and sizes, these siliceous spicules' morphology plays a pivotal role in assessing and understanding sponges' taxonomic diversity and evolution. In marine ecosystems, when sponges die their bodies disintegrate over time, but their spicules remain in the sediments as fossilized records that bear ample taxonomic information to reconstruct the evolution of sponge communities and sponge phylogeny.

Traditional methods of identifying spicules from core samples of marine sediments are labor-intensive and cannot scale to the scope needed for large analysis. Through the incorporation of high-throughput microscopy and deep learning, image classification has made significant strides toward automating the task of species recognition and taxonomic classification. Even with sparse training data and highly specific image domains, deep convolutional neural networks (DCNNs) were able to extract taxonomic features among morphologically diverse microfossils. Using transfer learning, training a classifier on pretrained DCNNs has achieved recent successes in classifying similar microfossils, such as diatom frustules and radiolarian skeletons.

In this project, I address the reliability of pretrained models to perform spicule identification and class-level classification. Using FlowCam technology to photograph individual microparticles, our dataset consists of spicule and non-spicule types without additional image segmentation and augmentation. Our proposed method is a pre-trained model with a custom classifier that performs two different binary classifications: a spicule vs non-spicule classification, and a taxonomic classification of Demospongiae vs. Hexactinellida. We evaluate the effect of implementing different DCNN architectures, data set sizes, and classifiers on image classification performance. Surprisingly, MobileNet, a relatively new and small architecture, showed the best performance while still being the most computationally efficient.

Other studies that didn't involve MobileNet had similar high accuracies for multi-class classifications with fewer training images. The reliability of DCNNs for binary spicule classification implicates the promising approach of a more nuanced multi-class/taxonomic classification. Future work should build multi-class classification that ranges more biogenic materials for the identification or more sponge taxonomic levels for species classification.

Keywords: Convoluted neural networks (CNN) · Sponge Spicules · Porifera · Microorganisms · Transfer learning · Deep Learning · FlowCAM · Feature extraction · SVM

ACKNOWLEDGEMENTS

I would like to give my thanks to my committee member Dr. Nada Attar for imparting her knowledge and expertise in computer vision. If it were not for CS lecturers like her, I would not have pursued an academic interest in this field of study during my master's program.

I would like to give special thanks to my other committee member, Dr. Amanda Kahn, and fellow graduate student, Sydney McDermott, at Moss Landing Marine Labs, for orchestrating the data gathering process of this project. This work would not have been possible without their timely and insightful contributions.

Finally, I would like to express my deep gratitude to my project advisor, Dr. Philip Heller, for his patience and enthusiastic encouragement to see this project come to fruition and for giving me the opportunity to contribute to Poriferal Vision.

Table of Contents

List of Figures

List of Tables

1. Introduction

Sponges (Phylum Porifera) are aquatic animals that come in a wide range of colors, shapes, and sizes. With approximately 8,550 species, sponges encompass four classes: Calcarea, Hexactinellida, Demospongiae, and Homoscleromorpha, of which Demospongiae is by far the most speciose, and Hexactinellida dominate in the deep sea [1]. Depending on a clade- or species-level taxon, their skeletal elements, namely spicules, are genetically unique in composition, shape, and size [2]. Their material composition is either calcium carbonate (a Calcarea characteristic) or silica (the remaining 3 classes) [1].

Numerous studies have focused on the assessment of sponge communities over time [3]. Acting as paleoenvironmental markers, siliceous spicules can help to reconstruct the geographic ranges of sponge taxa and infer new environmental conditions for a historical period [3]. This domain of research can range from how sponges play a role in the biogeochemical cycling of silica (due to their silica dependence) to discovering how their populations may have responded to past climate change events to help forecast global warming implications [3,4]. In recent years, the classification of demosponges and hexactinellids has gone through many modifications with various proposed morphological intermediates [5]. This difficulty of morphological classification originates in part from data gathering being too time-consuming and requiring thorough expertise beforehand. This is true of a variety of microfossils such as diatom frustules and radiolarian skeletons [6].

Given the laborious nature of taxonomic practices, state-of-the-art image classification has potential as a method for reliably extracting and classifying various morphology features. In particular, the recent application of deep convolutional neural networks (DCNNs) has shown

significant promise across various species-labeled datasets. Compared to previous traditional machine learning models, pre-trained models benefit from their ability to extrapolate new features from a sparse dataset. Known as transfer learning (TL), a pre-trained model from an image domain can be repurposed for a new problem+dataset [7].

In DCNN terminology, “feature extraction” refers to the process of transforming raw pixel values into preservable, higher-level image representations. After feature extraction, DCNNs can be configured for different types of classification problems. For taxonomic identification, Ahmed et al. achieved an accuracy of 96% on microscopic bacterial images using an Inception-V3 model and a support vector machine (SVM) [8]. Blaschko. et al. acquired accurate classification results through a similar method. Their image dataset consists of plankton species like diatoms and dinoflagellates [30]. As with their project’s image gathering, their plankton dataset was collected via an image segmentation hardware called FlowCam, which produces a digital photograph of each individual particle in a sample. Given DCNNs’ successes in similar image-based microscopic domains, the utilization of DL on spicule images is a promising approach.

Through model evaluation, this study seeks to demonstrate the reliability of DCNNs for spicule-based image recognition & taxonomic identification. This paper covers two binary classifications: a spicule to non-spicule (diatoms, radiolarians, inorganic sediments, etc.) approach, and taxonomic classification of Demospongiae vs. Hexactinellida. I evaluated 4 pre-trained model types through several evaluation protocols to assess their computational differences. These evaluation protocols used different dataset sizes, classifiers, and architecture variants. Different dataset sizes to derive what relevance the number of training images has on spicule identification and/or classification. In machine learning, there is no one classifier that is

always better than the others. Thus, this study makes statistical comparisons to know the best algorithm for its respective pre-trained model. Architecture variants were implemented to observe any disparity in either using an older or newer model version.To conclude, Section 7 discusses the generally high-performing variant of our proposed method and the possible workflow of future sponge taxonomists using DCNNs for multi-label sponge taxonomy.

2. DCNN for image classification

For many image classification problems, a typical CNN model architecture consists of a convolutional base, a pooling layer, and a classifier (Figure 1) [10]. Most of the network's computational load and user-specified parameters will be in the convolutional layer. This convolutional layer is designed to apply filter kernels to images or pre-existing feature maps. In essence, a feature map is the output of one filter applied to the previous layer. The layer's kernel-based filters introduce translation invariance and parameter sharing via convolutions. Translation invariance means that the system produces exactly the same response, regardless of how its input is shifted. Simply put, convolutions are an element-wise multiplication and summation of the input and kernel/filter elements.When feature maps are produced, padding is applied to ensure the output has the same size as the original image. This event highlights the presence of a feature in an image [10]. From the human perspective, this idea is akin to distinguishing a microorganism's body shape as either needle-like, rectangular or circular.

By typically performing max-pooling, the pooling layer reduces the image dimensionality by taking the maximum value for patches of a feature map and using it to create a downsampled (pooled) feature map [10]. This step preserves important information while shrinking the data [7]. As the architecture's last output, the classifier is usually composed of fully connected layers

(FCL) with a softmax function. FCL with a softmax function is explained in detail in Section 3. However, there are plenty of classifier alternatives that include global average pooling, SVMs, and other traditional machine learning classifiers. The classifiers used in this paper were selected based on their prevalence in other taxonomic studies. Other supervised machine learning algorithms like K-nearest neighbors, Random Forests, and Decision Trees should be explored but currently tend to underperform in taxonomic studies.

A CNN's main goal is to generate feature vectors to see whether an input image belongs to a particular class or another. As additional layers are added for more complex functions, more hidden layers/neurons deepen the CNN to recognize specialized features like complex shapes toward the end of the architecture [11]. However, the first few layers would focus on learning feature detectors like corners, edges, bends, etc.

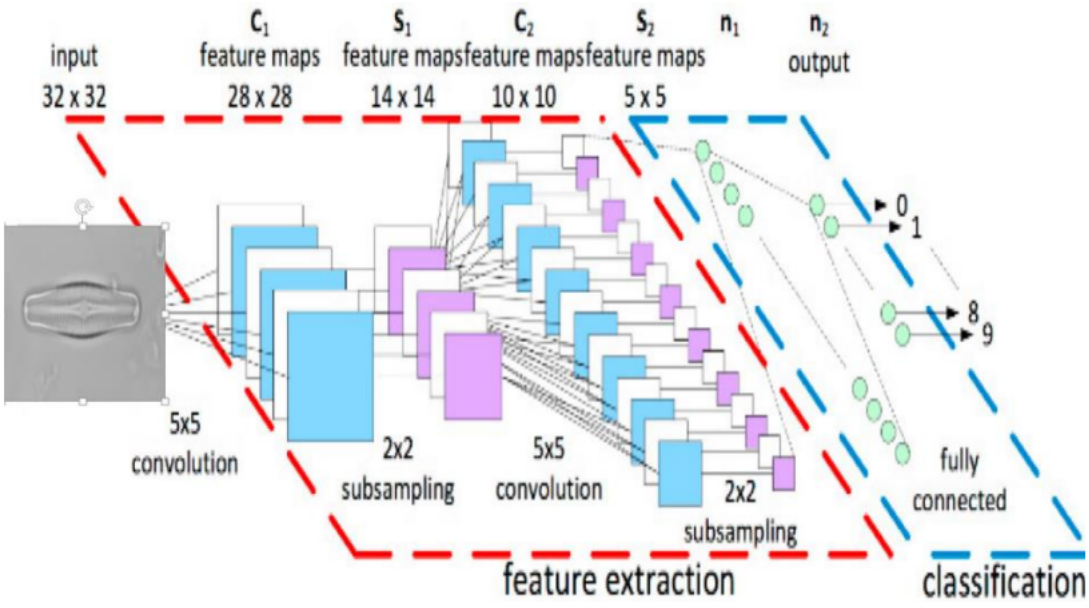


Figure 1: CNN Architecture

This architecture has the concept of hierarchical feature extractors. It learns highly abstract features from the diatom image and identifies its characteristics efficiently [7].

Deep learning (DL) models have an advantage in automatically learning hierarchical feature representations. This is because the general features of their first layer can be reused in different image domains in technical fields such as microscopy. Once applied, the last few layers concatenate already known features with new features specific to the task at hand [12]. Trained on larger image sets, DCNNs' concatenated features can facilitate the classification of spicules with sparse datasets. In turn, this alleviates the previous barriers to investing human resources to acquire large amounts of training data.

2.1. ML-based research in microorganism image recognition

ML-based methodologies have been applied to many different microbes for image analysis. Recent techniques have been implemented on different types of microorganisms including fungi, bacteria, algae, and protozoa (Figure 2). Since 2015, researchers have proposed hybrid systems based on CNNs for feature extraction and an SVM as their most accurate classifier (Figure 3)[9].

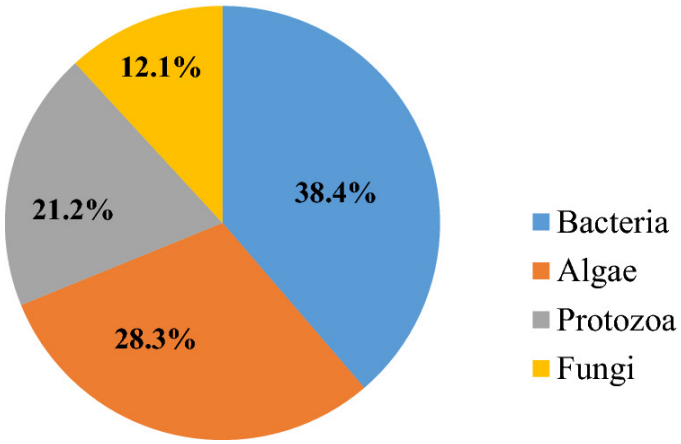


Figure 2: Impact of ML techniques on Microorganism Research

This pie chart spans 100 publications from the following online databases and digital libraries: IEEE Xplore, Science Direct, Springer Link, Google Scholar, ACM Digital library, and PubMed., Their yearly distribution was from 1995 to 2021 [9].

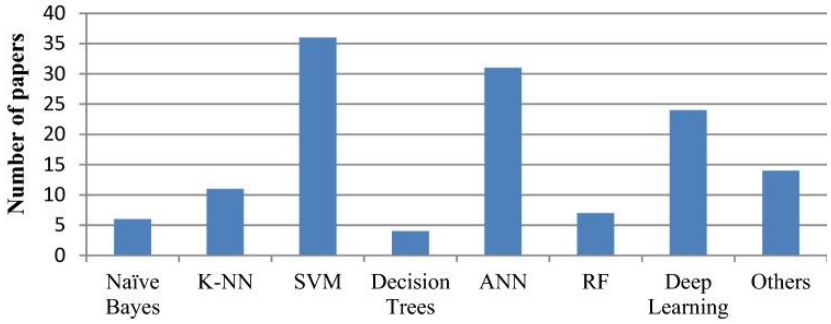


Figure 3: Distribution of 100 ML-based papers in Microorganism Research

This bar chart was reproduced from a meta-analysis of ML methods using the following search terms: “Microorganism classification” OR “Detection”, “Bacteria identification” OR “classification”, “algae”, “protozoa”, “fungi”, “ML”, “neural networks” “DL” [9]

3. Methods

This project will exhibit the 4 types of DCNNs described above for spicule image recognition and taxonomic classification. This group of pre-trained models is selected due to their applications in previous studies and the coexistence of newer versions amongst them. A convolutional base is generated from these specific models: VGG16, VGG19, ResNet50, ResNet152V2, InceptionV3, InceptionResNetV2, MobileNetV1 and MobileNetV2. With no potential complications, there are to be three 3 different classifiers for each base: Fully Connected Layers, Global Average Pooling + Sigmoid, and a Linear SVM.

The DCNN Architectures listed above are trained, validated, and tested for two main binary classifications: “NonSpicule or Spicule" (NS) and “Demospongiae or Hexactinellida" (DH). For file clarity, these classifications' nomenclature have followed this format of identification: “FirstClassInitial+SecondClassInitial_ModelName_datasetsize.ipynb”. Upon instantiating the convolutional base, we set our model parameters as the following:

- Weights to equal “ImageNet”
- Configure include-top as false for training our classifier
- A different input shape (image tensor shapes) akin to the pre-trained model’s defaults

Through a python script call, the data for our convolutional base is already divided into “train”, “validation”, and “test” folders with each image’s name as “ClassName (current #).png”. The flow_from_directory method automatically infers image labels from the directory structure to be fed into feature extraction [36]. At model compilation, we set the optimizer to Adam and the loss to Binary Cross Entropy (BCE). Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based on training data. We use the Adam optimizer because it tends to require fewer parameters for tuning and faster computation time. Binary cross entropy compares each of the predicted probabilities to the actual class output which can be either 0 or 1. It then calculates the score that penalizes the probabilities based on the distance from the expected value. That means how close or far from the actual value. For binary classification, the range of the output value is 0 to 1 when we pass it through a sigmoid activation instead of a softmax activation (Figure 4).

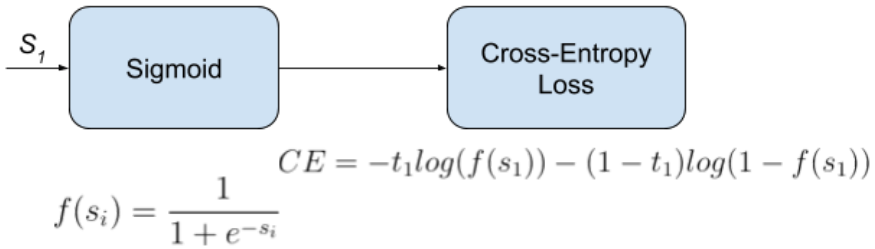


Figure 4: Binary-Cross Entropy Loss

This diagram sets up a binary classification problem between C” = 2 classes for every class in C. The long formula of Cross Entropy Loss is often used when using this loss. [37]

3.1. Convolutional base Models

The ImageNet Large Scale Visual Recognition Challenge, or ILSVRC for short, is an annual competition held between 2010 and 2017 in which challengers are tasked to use subsets of the ImageNet dataset. ImageNet is an image database organized according to the WordNet hierarchy (only applicable to nouns). Each node of the hierarchy is portrayed by hundreds and thousands of images. The subset of the ImageNet dataset, ILSVRC, has become the most popular subset of the dataset consisting of 1000 object classes to benchmark image classification algorithms. ILSVRC has resulted in a range of state-of-the-art DCNN models for image classification, the architectures and configurations of which have become heuristics and best practices in the field. The models used in this study still have their pretrained weights when the models were initially trained on the ImageNet dataset. There is a need for model evaluation because of the differences in model architecture and the number of parameters will likely have varying results in accuracy.

3.1.1 Visual Geometry Group Networks

Deep Learning-based taxonomic identification of bacteria and algae has recently been revolving around using Visual Geometry Group networks (VGG) for feature extraction. This CNN model uses only 3x3 convolutional layers and two fully-connected layers, each with 4,096 nodes followed by a softmax classifier (Figure A1, [13]). While still considered one of the most popular image recognition architectures for this practice, VGG networks are not as desirable as other smaller networks. A prominent reason is that VGG networks have model sizes of 533MB and 574MB for VGG16 & VGG19 respectively [14]. Large model sizes are time-consuming to implement, but the models' architecture outperforms previous models with its depth and number of fully-connected layers. In 2013, They rose to fame as the winning submissions of the

ILSVRC, surpassing GoogleLeNet by ~0.9% in test error [19]. Unlike other networks assessed in this study, VGG networks cannot be augmented with more layers without resulting in a vanishing gradient problem and increasing the training time significantly. The vanishing gradient problem when early layers are almost ignored in the learning process. Specifically, the network is unable to propagate the output's useful gradient information back to its earlier layers to tune their parameters. Thus, the network will not learn and prematurely converge to a poor solution. As the sensitivity to training error increases, adding layers may also result in a loss of accuracy due to performing optimizations on huge parameter space [16]. However, these networks have a defining trait of being a uniform architecture and straightforward to implement (Figure A2, [16]).

3.1.2. Residual Networks

When a network's depth increases, there are other non-accuracy issues to look out for while accounting for overfitting. Through the introduction of residual blocks, Residual Networks (ResNet) these issues like the vanishing gradient problem with identified mappings of two shortcut connection types: identity shortcuts and projection shortcuts. ResNet is a micro-architecture that is an accumulation of "building blocks" (standard CONV, POOL, etc. layers) (Figure A3, [17]). ResNet50 and ResNet152 are 50-layer and 152-layer deep CNNs, respectively. Comparing models to manual inspection, Mitra et al. applied ResNet50 Classification of six foraminifera species and comparative testing of the pre-trained architecture VGG16 [18]. For species classifications, both architectures were less sensitive/biased to specimen orientation than the humans' expert or novice selection. For visual comparison, Figure A4 shows an early layer synthesis between VGG19 and ResNet152, both of which mainly consists of 3x3 filters. When compared to VGG16 and VGG19, residual networks are

significantly smaller in model size (120 MB). This is attributed to the use of global average pooling (similar to GoogLeNet) instead of a fully connected layer [19].

3.1.3. Inception Networks and Inception-Residual Networks

In contrast to other pre-trained deep models, Inception networks are “wider” and heavily engineered to improve performance speed and accuracy [38]. Essentially, these networks are wider because they compute multiple different conversions in parallel and concatenate them into a simple output. Primarily, the Inception framework seeks to tackle common deep learning hurdles like overfitting and high computation expenses [38]. Overfitting means the training has focused on the particular training set so much that it has missed the point entirely. When this happens, the algorithm, unfortunately, cannot perform accurately against unseen data, defeating its purpose.With this in mind, Inception Network’s neural architecture is built with a dimension-reduced module that has kernels of multiple sizes operating on the same level. Initially called GoogLeNet, Inception-v1 had 9 linearly stacked modules and two auxiliary classifiers to also prevent the vanishing gradient problem [20].

Inception-v2 and Inception-v3 focused on updates to the inception module to further increase ImageNet classification accuracy and reduce computational complexity [21]. Inception-v3 updates the following: the batch norm in the auxiliary classifiers, label smoothing, and factorized 7x7 convolutions [21]. Coming in at 92MB, the network’s weights are smaller than previously shown VGG weights and ResNet weights (Figure A5, [21]). Using the Environmental Microorganism dataset, Liang et al. optimized this network for image classification via fine-tuning its dropout rate and neuron number to yield a 92.9% accuracy [22].

InceptionResNetV2 (215 MB) has a hybrid inception module that simulates the performance

of ResNet by introducing residual connections [23]. At its conception, this Inception-v3 variant was significantly deeper, thus being more accurate than its previous state-of-art models. InceptionResNetV2 has its inception blocks recast, with fewer parallel towers than its predecessor Inception-v3 (Figure A6, [24]).

3.1.4. MobileNet & MobileNetV2

MobileNet networks are particularly useful for mobile and embedded vision applications due to applying the concept of Depthwise Separable Convolution (a depthwise convolution followed by a pointwise convolution). These networks have a lighter computational complexity, and smaller model size (17 or 14 MB). Their architecture's computation reduction stems from applying Batch Normalization and Relu after each depthwise separable convolution (Figure A7, [-]). Compared to Visual Geometry Group and Inception Networks' convolutions, this type of convolution results in only a 1% loss in accuracy with significant increases in computational efficiency. This difference in computation is also due to compiling fewer multiple additions (multiply and add calls) and parameters. In general, MobileNet networks are at least up to par with networks like VGGNet and Inception-v3 (which were 1st Runner Up of ILSVRC 2014 & 1st Runner respectively in ILSVRC 2015) [26].

No image classifications using MobileNet networks have been applied to microorganisms to date; however, these networks are noteworthy to include because of their computational efficiency. MobileNetV2 has a noteworthy rendition of two types of blocks with 3 distinct layers. With this architecture, MobileNetV2 could outperform MobileNetV1 in model size, computational cost, and inference time (Figure A8, 35).

3.2. Classifiers

3.2.1 Fully-connected layers (FCL)

To avoid the model from overfitting, FCL classifiers were built with 4 layers: a flattening layer, 2 dense layers, and a dropout layer. Through a flattening layer, the feature maps' multidimensional output is configured to a single long feature vector. Succeeding the flatten layer, two dense layers ensure every neuron receives inputs from all the neurons of the previous layer. Finally, we also used a dropout rate of 50% that randomly discards some sets of neurons between the dense layers. Overall, Fully Connected layers in neural networks are those layers where all the inputs from one layer are connected to every activation unit of the next layer.

3.2.2. Global average pooling (GAP)

As a more extreme type of dimensionality reduction, GAP was made to potentially replace a standard stack of fully-connected layers that are still overfitting. We embed a global average pooling layer that takes the average of each feature map as the resulting vector. As proposed by Lin et al., this strategy explicitly enforces feature maps as confidence maps of the intended categories [39]. In general, a confidence map is a probability density function on the new image, assigning each pixel of the new image a probability, which is the probability of the pixel color occurring in the object in the previous image.With the absence of optimizable parameters, overfitting is less likely to occur. Instead of the softmax function, the sigmoid function is included as an additional parameter to the activated layer for better binary classification.

3.2.3. Linear support vector machines (SVM)

A simple linear SVM finds a hyper-plane that creates a boundary between the types of data but this hyperplane is a line in 2-dimensional space. That means all of the data points on one side of the line will represent a category and the data points on the other side of the line will be put into a different category. While our FCL and GCP classifiers both use hold-out validation, our linear SVM classifier uses five-fold cross-validation to estimate the error of the classification [40]. Hold-out validation has the dataset split into 'train' and 'test' sets while cross-validation has the dataset randomly split into 'k' groups. Each 'k' group becomes the 'test' set while the rest are used in training. While regarded to be more supportive of insight for performance on unseen data than hold-out validation, cross-validation is expected to also have more computational power and runtime [41]. Tao et al. implemented threefold cross-validation on a linear SVM to classify six known algal species and two unknown algae. The proposed method exhibits better performance than alternative methods such as the K-NN classifier and radial basis function-based SVM [28].

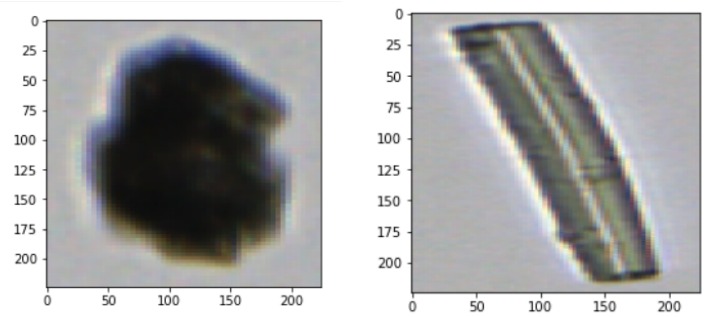
4. Data Preparation & Experiments

4.1 Data Description

This study used sediment core samples collected during the International Ocean Discovery Program Expedition 323 to the Bering Sea (IODP Exp 323). Provided by the Aiello research group at Moss Landing Marine Laboratories (MLML), these Bering Sea sediment samples date back to the Pliocene warm period. This period includes global climate events like the Northern Hemisphere glaciation and glacial-interglacial climate cycles [29]. Four exemplary samples were

sent to Nicole Gill (at Yokogawa Fluid Imaging Technologies, Inc.) to be run at 4x and 10x magnification levels on their automated particle analysis instrument, the FlowCam. In our data gathering, larger spicule types ("megascleres") are visible via the instrument's 4x objective while smaller spicules ("microscleres") are visible in its 10x objective. Individual particles, such as biogenic materials, were also imaged in the flow of water sediments and were considered a non-spicule dataset for the project. Dr. Amanda Kahn of MLML assessed and supervised the FlowCam methods of 4x/600 FOV, 4x/300 FOV, and 10x/100 FOV (here FOV refers to "field of view" - the depth of field of the flow cells used at each magnification). Subsequently, Sydney McDermott, a graduate student in the MLML Invertebrate Ecology Lab, performed image preprocessing and categorized the directory structure of produced FlowCam images by filter scheme, non-spicule type, and spicule type.

For the purposes of this project, 10x focused spicule/nonspicule datasets are used as shown in Table I. As for image preprocessing, all images were filtered by edge gradient (a measure of fuzziness), filtered on compactness ("fluffy" particles were removed), and finally filtered based on circularity (how round or elongated they are). Within this imageset, the spicule images clearly identifiable as Demospongiae or Hexactinellida are considered exemplary based on their image quality (Figure 4). Since each FlowCam image consists of a single particle, no additional segmentation protocols were needed for image pre-processing.



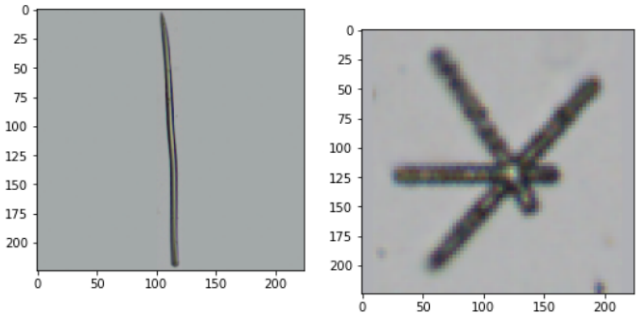


Figure 5: Nonspicule & Spicule, Spicules of Demospongiae & Hexactinellida
(top two, and bottom two respectively)

No feature vectors around differential, contour representation, moment, texture, and shape were fed as input to the pre-trained models in this project. In particular, contour detection and grayscale, which were found to be beneficial for algal image classification studies, were excluded from the experiments but were used instead for image pre-processing [10]. Similar to the ML-based approach proposed by Blaschko et al., our project aims to primarily achieve satisfactory classification results by implementing mixed, DCNN-derived features with a simple ML algorithm [30].

TABLE I. OVERALL 10x FOV SAMPLE 1 DISTRIBUTION OF DATASET

Dataset	**Biogenic Material Type**	Total Number of Images
Positive	Spicules	2205
Negative	NonSpicules (ie diatoms and radiolarians)	33644
Spicule Class	Demospongiae	3481
Spicule Class	Hexactinellida	1377
Total	-	40707

4.2. Training-Test-Validation Split (TTV)

For non-spicules or spicule classification, I first ran a small dataset of 400 images that splits into training, testing, and validation sets for the input of a designated pre-trained model. The complete dataset of exemplar spicules/non-spicules was then trained following each small data set trail run. The small dataset had a 50-25-25 percent TTV split. For Demospongiae vs Hexactinellida classification, Tables 3 and 4 show the complete dataset distribution of a 70-15-15 percent TTV split (Table 2).

TABLE II. SPICULE IDENTIFICATION SMALL DATASET DISTRIBUTION

Dataset	**Biogenic Material Type**	**Training Images**	**Testing Images**	**Validation Images**
Positive	Spicules	100	50	50
Negative	NonSpicules (ie diatoms and radiolarians)	100	50	50
Total		200	100	100

TABLE III. SPICULE IDENTIFICATION COMPLETE DATASET DISTRIBUTION

Dataset	**Biogenic Material Type**	**Training Images**	**Testing Images**	**Validation Images**
Positive	Spicules	1542	332	331
Negative	NonSpicules (ie diatoms and radiolarians)	23549	5047	5048
Total		25091	5379	5379

TABLE IV. DEMOSPONGIAE & HEXACTINELLIDA DATASET DISTRIBUTION

Dataset	**Biogenic Material Type**	**Training Images**	**Testing Images**	**Validation Images**
Sponge Class	Demospongiae	2435	524	522
Sponge Class	Hexactinellida	962	207	208
Total	-	3397	730	731

4.3. Metrics for Model Evaluation

Comparisons of the four different models were based on six performance metrics. Primarily, testing accuracy on unseen data is calculated with the number of correctly classified images over the number of all classified images. Computation cost and evaluation protocol runtime differences were also observed with each type of accuracy. Neural network depth and a number of epochs were also charted with each experiment run to numerically compare computation complexity and training time.

The amount of overfitting in our models can be tracked with training/validation accuracies and losses. When the fluctuating accuracies are charted on the accuracy vs epoch scale, the size of the gap between the maximum and minimum shows how much overfitting might be taking place. As shown in Figure 5, the green validation accuracy curve illustrates good tracking and the training accuracy avoids overfitting. Given that this project is primarily about model evaluation, evaluating how our DL models adapt to new training/validation data is critical for proper performance checks. Hence, training/validation losses are plotted together to assess a model's ability to fit the training/validation data. Any divergence from the plotting of Figure 5's Training/validation losses suggests underfitting or overfitting.

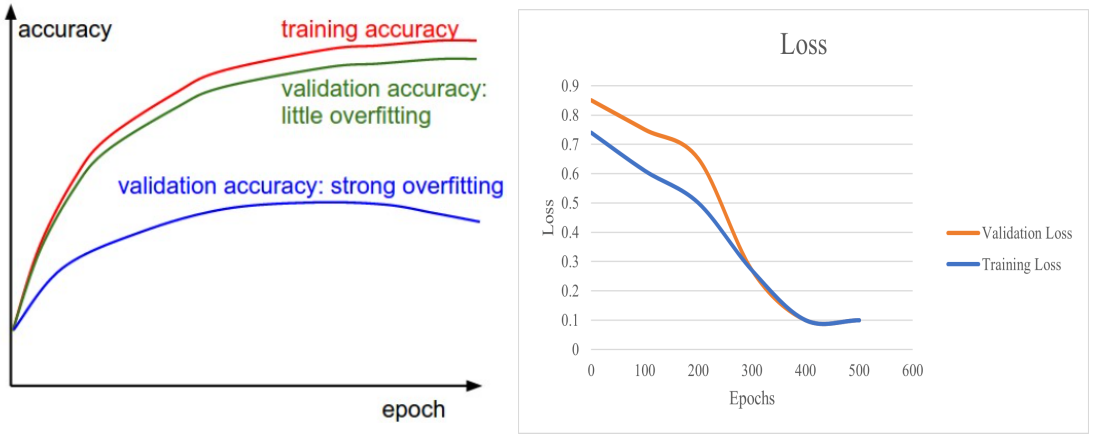


Figure 6: Training/Validation accuracy [31] & Training/Validation loss plots [32]

5. Results

5.1. Spicule Identification (SI)

For Spicule Identification, Table 5 summarizes the cross-validated accuracy scores of applying SVMs to the pretrained models' extracted features when using the small dataset of spicules and non-spicules. There are no metric-based differences when training, testing, and validating with only 400 images on a 50-25-25 TTV split. Training accuracies were near 100% across the board but models that used SVM and cross-validation took more epochs to reach the same level of accuracy when evaluated against the models that did not use SVM and cross-validation.

TABLE V. SI SMALL SVM CLASSIFICATION TEST RESULTS (FCL, GAP EXEMPTED)

Pretrained Model	Accuracy	Standard deviation
VGG16	0.99	0.01
VGG19	0.99	0.01
ResNet50	1.00	0.00
ResNet152V2	1.00	0.00
Inception-v3	0.99	0.01
InceptionResNetV2	1.00	0.00
MobileNetV1	1.00	0.00
MobileNetV2	1.00	0.00

Using the complete dataset, SI models with classifiers of FCL and GAP showed no apparent accuracy differences when running through the training, validation, and test datasets (Table 6 & 7). Aside from certain ResNet outputs, test accuracies were all above 95% for all experiments while corresponding training accuracies were near 100% (Figures 7 & 8). This extends to validation accuracies not straying far from the training accuracies when plotted for possible overfitting. Though, ResNet50 was the exception to this observation.

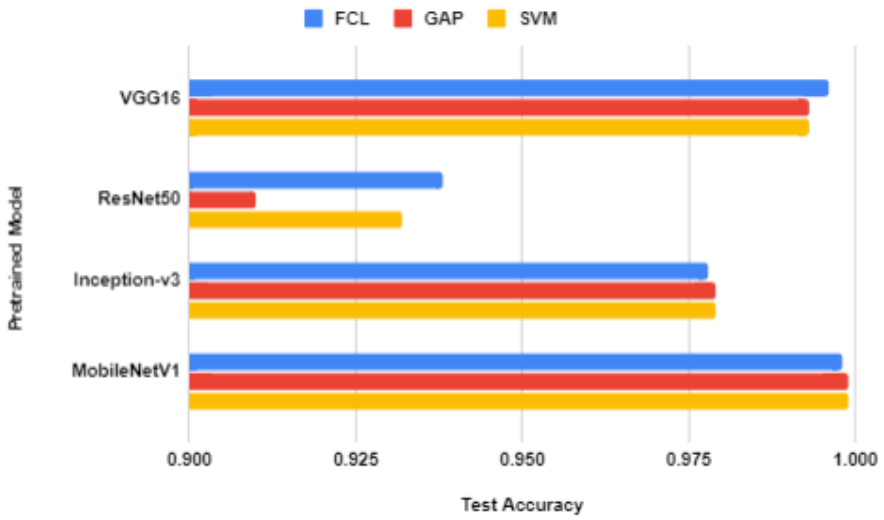


Figure 7: Test Accuracies of Spicule Identification on Older models

TABLE VI. SI ON COMPLETE DATASET OF OLDER VERSIONS

Pretrained Model	**Classifier**	**# of epochs**	**Test Accuracy**	**Training Accuracy**	**Epoch w/ Earliest Highest Training Accuracy**
VGG16	FCL	20	.996	1.0	9
	GAP	100	.993	.9980	73
	SVM	-	.993	.99	-
ResNet50	FCL	20	.938	.7168	3
	GAP	50	.910	.8524	43
	SVM	-	.932	.99	-
Inception-v3	FCL	100	.978	1.0	83
	GAP	100	.979	1.0	3
	SVM	-	.979	.99	-
MobileNetV1	FCL	20	.998	1.0	6
	GAP	10	.999	1.0	8
	SVM	-	.999	.99	-

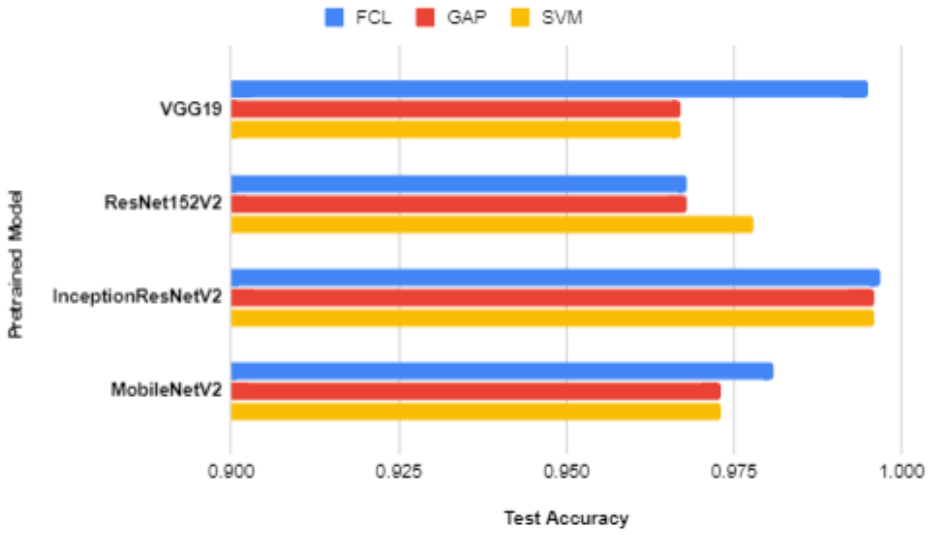


Figure 8: Test Accuracies of Spicule Identification on Newer models

TABLE VII. SI COMPLETE OF NEWER VERSIONS

Pretrained Model	**Classifier**	**# of epochs**	**Test Accuracy**	**Training Accuracy**	**Epoch w/ Earliest Highest Training Accuracy**
VGG19	FCL	20	.995	1.0	8
	GAP	100	.967	.9900	50
	SVM	-	.967	.99	-
ResNet152V2	FCL	100	.968	1.0	5
	GAP	100	.968	1.0	2
	SVM	-	.978	.99	-
InceptionResNetV2	FCL	20	.997	1.0	3
	GAP	100	.996	1.0	3
	SVM	-	.996	.99	-
MobileNetV2	FCL	20	.981	0.99	4
	GAP	100	.973	1.0	4
	SVM	-	.973	.99	-

5.2. Demospongiae vs Hexactinellida (DH) Classification

Though not mentioned in the previous section, we did try to run DH classifications using a small dataset but concluded to discard these experiments after several poor model performances (data not shown). The models were unable to increase training/validation accuracy past 30%. Irrespective of the DCNN+classifier model involved, there were not enough images to accurately extrapolate enough distinct features. Given Demospongiae's plethora of spicule morphologies against those of Hexactinellida, the amount of training data heavily influenced how well a model extracts enough features to distinguish one class from another.

Regardless of the model components, spicule classification (DH) models with complete datasets had mostly high training/validation accuracies. As shown in Table 8 & 9, testing accuracies were above 90% for all models and classifiers, with the lowest average of the 3 classifiers being ResNet50 and the highest being MobileNetV2. SVM-based models had significantly highest cross-validated accuracies of 99%. The k-folds for this cross-validation only included training and validation datasets.

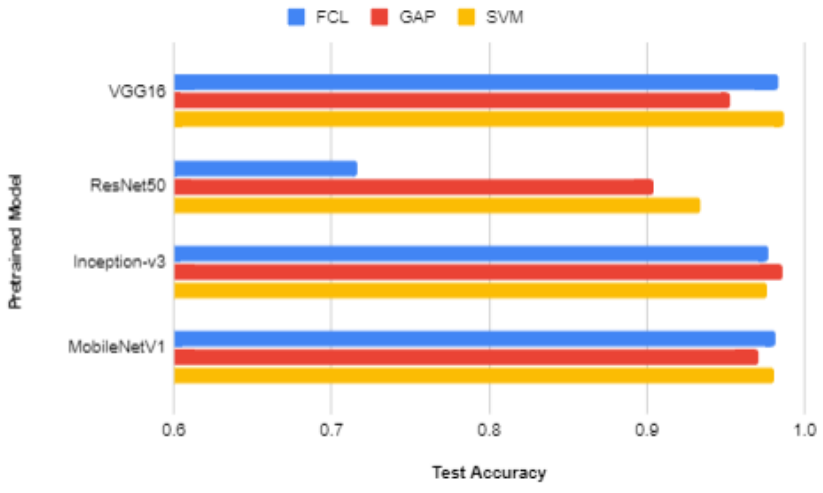


Figure 6: Test Accuracies of DH Classification on Older models

TABLE VIII. DH CLASSIFICATION ON OLDER VERSIONS

Pretrained Model	Classifier	# of epochs	Test Accuracy	Training Accuracy	Epoch w/ Earliest Highest Training Accuracy
VGG16	FCL	15	.983	1	6
	GAP	15	.953	1	15
	SVM	-	.987	.99	-
ResNet50	FCL	20	.716	0.9	3
	GAP	100	.904	0.87	87
	SVM	-	.934	.99	-
Inception-v3	FCL	15	.977	0.9	15
	GAP	15	.986	0.94	8
	SVM	-	.976	.99	-
MobileNet	FCL	15	.982	1	11
	GAP	15	.971	1	15
	SVM	-	.981	.99	-

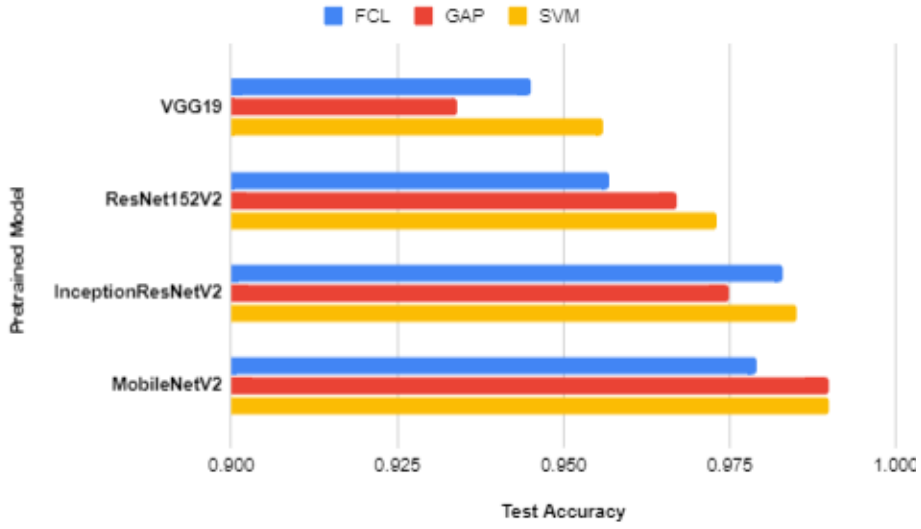


Figure 7: Test Accuracies of DH Classification Newer models

TABLE IX. DH CLASSIFICATION ON NEWER VERSIONS

Pretrained Model	Classifier	# of epochs	Test Accuracy	Training Accuracy	Epoch w/ Earliest Highest Training Accuracy
VGG19	FCL	20	.945	1	4
	GAP	100	.934	1	17
	SVM	-	.956	.99	-
ResNet152V2	FCL	20	.957	1	12
	GAP	20	.967	1	14
	SVM	-	.973	.99	-
InceptionResNetV2	FCL	20	.983	0.96	15
	GAP	100	.975	0.94	4
	SVM	-	.985	.99	-
MobileNetV2	FCL	15	.979	1	11
	GAP	15	.990	1	15
	SVM	-	.990	.99	-

6. Discussion

In this paper, this deep learning methodology circumvented the traditional need for feature descriptors and image augmentation to generate adequate training data. Owing to our near 99% correct classifications for most of our models, there was no need for additional fine-tuning for possible accuracy gains. While models like ResNet50+GAP could benefit from fine-tuning, the time investment to compute unique parameter optimization through additional experiments still makes it not a preferable model for real-time application.

Despite this complete dataset being smaller than the Imagenet dataset (with 1000 classes),

our dataset size was still sufficient for binary classifications. However, the equally high classification accuracies of our small dataset came from how deep and complex the pre-trained models were compared to traditional ML models. In addition to other animals' complex shapes, ImageNet weights can reliably generate vital feature descriptors such as the contour, perimeter, area, and mean pixel intensity of microbes. Another cause can be the high visual quality of the FlowCAM images of both spicule and non-spicule types. In the future, an image's composition could have indistinguishable shapes of microorganisms when it includes unwanted artifacts and low resolution/focal depth. In experiments around this possible concern, data quality correlations should be further explored based on less focused and exemplary images.

6.1 Influence of dataset reduction

Using a smaller dataset did not significantly affect the overall accuracy of spicule identification. Aside from needing more epochs to reach maximum accuracy, SI small-data models with FLC or GAP ran in much shorter runtimes compared to running the SVM Classifier (20 minutes vs 40 minutes on average). As previously mentioned in related studies, DCNN+SVM accuracy became a reliable DL technique to perform well on sparse datasets. For example, Arredondo-Santoyo et al. shown the ResNet-C-SVM combination has the best accuracy on an imbalanced dataset of 1024 fungal assay images [33]. They were able to overcome the class imbalance problem and overfitting problem through image augmentation and Synthetic Minority Over-sampling Technique (SMOTE) [33]. Further image augmentation did not seem necessary in our case due to the precedent of our spicule identification being based on binary classifiers. This is subject to change for future closely related levels of taxonomic complexity come to play with binary/multi-class distinctions. Aside from those of residual networks, test and

cross-validation accuracies of over 90% give a good indication of how well our models performed on data exempted from training or validation upon compiling.

For taxonomic classification, we often needed to use all available images for good performance metrics. Nonetheless, the computational complexity and runtime differences for building this task's SVM classifiers were much larger than their FCL and GAP counterparts. While FCL and GAP were computed approximately within 2 hours, SVM classifiers need more than ten gigabytes of memory and ran for five more hours to build their model and commute their cross-validated accuracies. When performing the final model evaluation, this disparity was enough to deduce SVM-based models as the most expensive classifiers to compute. Irrespective of other taxonomic studies' reliability of SVMs, the higher performance of other classifiers with various pre-trained models and more computational expense have now led us to eliminate SVMs in our final model selection.

6.2. Comparative Analysis of Different Pretrained Models

Within our model evaluation, the relationship between computational complexity and runtime performance is best observed by how efficient the DCNNs were in automatically extracting features. MobileNet neural networks were the fastest of the 4 pretrained model types regardless of the classification, classifier (SVM, FCL, GAP), and data size. This response was due to their feature extraction and training stages being significantly faster. Irrespective of their architectural differences, MobileNetV1 and MobileNet2 had insignificant differences in performance and accuracy (by only 1%).

The ResNet networks were by far the slowest, least accurate, and computationally heavy

models to train, regardless of the classification application (SI or TC). As shown in Tables 7 & 9, global average pooling does exhibit some ResNet50 accuracy improvements, but at the cost of more epochs and longer runtimes. With a significant improvement over ResNet50, ResNet152V2 had nearly perfect accuracy on most evaluations but was still significantly slower than VGG, MobileNet, and Inception networks.

VGG16 and Inception-v3 had a near inverse relationship in the overall model evaluation due to the differences in how they tackled feature extraction and training. After the ResNet model, the VGG16 model was the second slowest in feature extraction but trained faster when using FCL. Proving the inverse relationship between them, Inception-v3 trained less efficiently by needing higher epochs but was faster than VGG16 in extracting features. As shown in Table 9, the accuracy of using VGG19 could improve if the VGG-FCL is implemented. The table also exhibits Inception-FCL models taking a significant amount of training time to reach 97% accuracy when compared to their other Inception counterparts. This was positively correlated with the reduction in training time when using InceptionResNetV2. However, InceptionResNetV2 took more time to extract features than other Inception networks.

To conclude, while some models were slower, all had an accuracy of at least 90% demonstrating that further image preprocessing was not needed at this stage of SI and SC. However, the selection of only 3 classifiers does limit the scope of classification error. If a multiclass classification is done later, there may be more false negatives and false positives between closely-correlated classes. When it came to MobileNet and ResNet networks, the depth and model size still played a role in model evaluation but not as much when it's between VGG networks vs Inception networks. In Table 5 & 6's, 100 epochs was a default epoch number for bigger networks but weren't necessary for most cases of Earliest Highest Accuracy where we

received desired training and validation accuracies on specific epochs. Based on the metrics shown in this study, MobileNet+Any Classifier and MobileNet+GAP were the best models for spicule identification and taxonomic classification respectively.

6.3 Performance Metrics

Given the high performance of the image recognition and classification tasks, numerical performance metrics like accuracies and losses did not shed much light on additional model evaluation. This may stem from the prior image segmentation and collection outputting FlowCam images of high quality and quantity. These images exclude other artifacts in each microscopic image. The high performance on our FlowCam images was consistent with other FlowCam studies that used supervised learning approaches like single classifiers but their own segmentation procedure [30]. Ranging from 1% to 3%, some test accuracy gains were achieved when using FCL over the other classifiers when it came to spicule identification.

As shown in Figure 5, these DL experiments had a very similar relationship between the accuracies and losses of training/validation sets to further reassure the TL models' performance were not overfitting. Corresponding test accuracies and cross-validated accuracy scores demonstrate how reliably the models performed on unseen data and how well the model was enriched given ample training data. This fact extended to all classification tasks, 4 DCNN architectures, and subsequent classifier types accordingly.

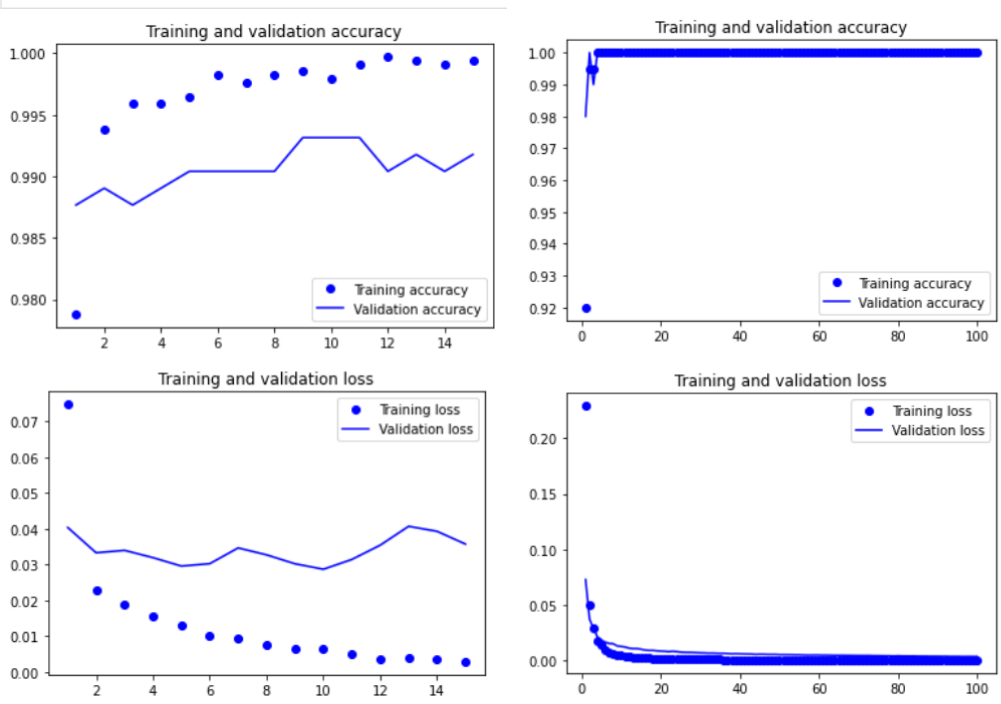


Figure 6: 4 Training/Validation Accuracies DH Classification & SI Classification

Taken from our Mobilenet+GAP experiment (left) & Mobilenet+GAP experiment (right)

7. Conclusion

In this paper, we demonstrate 4 DCNN architectures as convolutional bases for extracting image features for spicules recognition and classification. To keep the procedure simple and efficient, we employ transfer learning by implementing pre-existing ImageNet weights before our own ML classifier. FCL, GAP, and SVM are the traditional classifiers used in the experiments to systematically test overall architectural differences. FCL and GAP-orientated models had mostly insignificant trade-offs depending on the convolutional base. Meanwhile, SVM-orientated models had problems running on smaller datasets and had longer training times. These SVM issues are subject to change with future improvements like better parameter optimization, larger datasets with more categorical features, and performing PCA. As for computational runtime, these experiments illustrate the importance of data size for taxonomic

identification and how reliable the pre-trained weights were for image recognition.

With high-performance metrics across the board, there were no major improvements in using newer ImageNet architectures over their predecessors (except, notably, InceptionResNetV2 and ResNet152V2 in certain cases). In both spicule recognition and classification, most ImageNet-pretrained DCNNs exhibit great classification performance. The MobileNet network has the best computational efficiency, showing great potential for real-time in-vehicle analysis of sponge spicules. Generally, FCL came out to be the most accurate classifier in most experiments but GAP had some higher accuracies for certain pre-trained models.

Our general model winner, MobileNet with GAP, can be customized for multiclass classification if provided with a large number of spicules images identified at the taxonomic level of order, family, genus, or species. As shown in many taxonomic studies on plankton, a large quantity of training data is not easily obtainable and available. This issue is due to the low number of available taxonomic specialists needed to collect and label enough images for each species for training purposes. To prevent this from affecting our classification accuracy, future classification tasks using pre-trained DL models may still rely on image preprocessing. Like our experiments, Michael et al. obtained similar results of high classification accuracy for diatoms using VGG16 with an FCL 256 neuron layer and a softmax classification layer. They instead annotated their input images with object contours using SHERPA, a diatom morphometric software. They performed this preprocessing around virtual slide scans [7] along with their own segmentation model like our FlowCam approach. Unlike our binary classification task, they train a classifier to classify 10 species-level categories, each with relatively few (100-300) images [7]. With limited investment in data collection, SHERPA has the key for later multiclass classifications to preprocess spicule images for classification improvements.

References

[1] El-Bawab, Fatma. “Phylum Porifera.” Essay. In *Invertebrate Embryology and Reproduction*, 106–69. London, United Kingdom: Academic Press, 2020.

[2] Hooper, John N.A. “Sponguide: Guide to Sponge Collection and Identification.” Sponguide: Guide to Sponge Collection and Identification. Queensland Museum, August 1, 2000. https://www.researchgate.net/publication/242495363_Sponguide_Guide_to_Sponge_Collection_and_Identification.

[3]Łukowiak, Magdalena. “Utilizing Sponge Spicules in Taxonomic, Ecological and Environmental Reconstructions: A Review.” PeerJ. PeerJ Inc., December 18, 2020. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7751429/.

[4]J. W. F. Chu, M. Maldonado, G. Yahel, and S. P. Leys. “Glass sponge reefs as a silicon sink,” Marine Ecology Progress Series 441:1-14. 2011.

[5] Botting, Joseph P., Yuandong Zhang, and Lucy A. Muir. “Discovery of Missing Link between Demosponges and Hexactinellids Confirms Palaeontological Model of Sponge Evolution.” Nature News. Nature Publishing Group, July 13, 2017. https://www.nature.com/articles/s41598-017-05604-6.

[6] Kloster, Michael, Daniel Langenkämper, Martin Zurowietz, Bánk Beszteri, and Tim W. Nattkemper. “Deep Learning-Based Diatom Taxonomy on Virtual Slides.” Nature News. Nature Publishing Group, September 2, 2020. https://www.nature.com/articles/s41598-020-71165-w.

[7] Chaushevska, Marija, Ivica Dimitrovski, Saso Dzeroski, and Hristijan Gjoreski. “Hierarchical Classification of Diatom Images with Transfer Learning.” Repository of UKIM, September 24, 2020. https://repository.ukim.mk/handle/20.500.12188/9475?mode=full.

[8] Ahmed T, Wahid MF, Hasan MJ (2019) Combining deep convolutional neural network with support vector machine to classify microscopic bacteria images. In: 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), Cox'sBazar, Bangladesh, pp. 1–5, 10.1109/ECACE.2019.8679397 [[Reflist](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8505783/#CR61)]

[9] Rani, Priya, Shallu Kotwal, Jatinder Manhas, Vinod Sharma, and Sparsh Sharma. “Machine Learning and Deep Learning Based Computational Approaches in Automatic Microorganisms Image Recognition: Methodologies, Challenges, and Developments.” Archives of computational methods in engineering: state of the art reviews. Springer Netherlands, 2022. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8405717/.

[12] Stewart, Matthew. “Simple Introduction to Convolutional Neural Networks.” Medium. Towards Data Science, July 29, 2020. https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac.

[13] Mishra, Mayank. “Convolutional Neural Networks, Explained.” Medium. Towards Data Science, September 2, 2020. https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939.

[14] Rout, Aparna R., and Sahebrao B. Bagal. “A Deep Learning Model for Image Classification.” https://www.irjet.in/. International Research Journal of Engineering and Technology, May 2017. https://1library.net/document/y8165d0z-a-deep-learning-model-for-image-classification.html#fulltext-content.

[15] Simonyan, Karen, and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition.” arXiv.org, April 10, 2015. https://arxiv.org/abs/1409.1556.

[16] Rosebrock, Adrian. “ImageNet: Vggnet, ResNet, Inception, and Xception with Keras.” PyImageSearch. pyimagesearch, June 17, 2021. https://pyimagesearch.com/2017/03/20/imagenet-vggnet-resnet-inception-xception-keras/.

[17] Vijayalakshmi A, Rajesh Kanna B. Deep Learning approach to detect malaria from microscopic images. *Multimed Tools Appl.* 2019;79(21–22):15297–15317. doi: 10.1007/s11042-019-7162-y.

[16] Koustubh. “ResNet, Alexnet, Vggnet, Inception: Understanding Various Architectures of Convolutional Networks.” CV-Tricks, August 9, 2017. https://cv-tricks.com/cnn/understand-resnet-alexnet-vgg-inception/.

[17] Poudel, Sahadev, Yoon Jae Kim, Duc My Vo, and Sang-Woong Lee. “Colorectal Disease Classification Using Efficiently Scaled Dilation in Convolutional Neural Network.” *IEEE Access* 8 (May 2020): 99227–38. https://doi.org/10.1109/access.2020.2996770.

[18] Mitra, R., T.M. Marchitto, Q. Ge, B. Zhong, B. Kanakiya, M.S. Cook, J.S. Fehrenbacher, J.D. Ortiz, A. Tripati, and E. Lobaton. “Automated Species-Level Identification of Planktic Foraminifera Using Convolutional Neural Networks, with Comparison to Human Performance.” Marine Micropaleontology. Elsevier, January 25, 2019. https://www.sciencedirect.com/science/article/pii/S0377839818301105.

[19] Boesch, Gaudenz. “VGG Very Deep Convolutional Networks (Vggnet) - What You Need to Know.” viso.ai, December 5, 2021. https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/.

[20] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. “Going Deeper with Convolutions.” arXiv.org, September 17, 2014. https://arxiv.org/abs/1409.4842v1.

[21] Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. “Rethinking the Inception Architecture for Computer Vision.” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, December 11, 2015. https://doi.org/10.1109/cvpr.2016.308.

[22] Liang, Chih-Ming, Chun-Chi Lai, Szu-Hong Wang, and Yu-Hao Lin. “Environmental Microorganism Classification Using Optimized Deep Learning Model - Environmental Science and Pollution Research.” SpringerLink. Springer Berlin Heidelberg, February 22, 2021. https://link.springer.com/article/10.1007/s11356-021-13010-9.

[23] Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. “Inception-V4, Inception-Resnet and the Impact of Residual Connections on Learning.” arXiv.org, August 23, 2016. https://arxiv.org/abs/1602.07261.

[24] Mehta, Parmita, Aaron Lee, Cecilia Lee, Magdalena Balazinska, and Ariel Rokem. “Inception Resnet V2 Architecture | Download Scientific Diagram.” https://www.researchgate.net/, May 7, 2018. https://www.researchgate.net/figure/Inception-Resnet-V2-Architecture_fig1_325015329.

[25] Tsang, Sik-Ho. “Review: MobileNetV1 - Depthwise Separable Convolution (Light Weight Model).” Medium. Towards Data Science, February 17, 2021. https://towardsdatascience.com/review-mobilenetv1-depthwise-separable-convolution-light-weight-model-a382df364b69.

[26] Tsang, Sik-Ho. “Review: MOBILENETV2 - Light Weight Model (Image Classification).” Medium. Towards Data Science, August 1, 2019. https://towardsdatascience.com/review-mobilenetv2-light-weight-model-image-classification-8febb490e61c.

[27]Liu, Li, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. “Deep Learning for Generic Object Detection: A Survey - International Journal of Computer Vision.” SpringerLink. Springer US, October 31, 2019. https://link.springer.com/article/10.1007/s11263-019-01247-4.

[28] Tao J, Chen W, Wang B, Jiezhen X, Nianzhi J, Luo T (2008) Real-time red tide algae classification using naive bayes classifier and SVM. In: 2008 2nd international conference on bioinformatics and biomedical engineering, pp 2888–2891. 10.1109/ICBBE.2008.1054

[29] Ivano W. Aiello, A. Christina Ravelo; Evolution of marine sedimentation in the Bering Sea since the Pliocene. Geosphere 2012;; 8 (6): 1231–1253. doi: https://doi.org/10.1130/GES00710.1

[30] Blaschko MB et al (2005) Automatic in situ identification of plankton. In: 2005 seventh IEEE workshops on applications of computer vision (WACV/MOTION'05), vol 1, pp 79–86. 10.1109/ACVMOT.2005.29

[31] Li, Fei-Fei, Jiajun Wu, and Ruohan Gao. “Babysitting the Learning Process.” *Neural Networks Part 3: Learning and Evaluation*. Reading presented at the CS231n: Deep Learning for Computer Vision Stanford - Spring 2022, May 10, 2022.

[32] Tokuç, A. Aylin. “Underfitting and Overfitting in Machine Learning.” Baeldung on Computer Science, January 1, 2022. https://www.baeldung.com/cs/ml-underfitting-overfitting#what-are-underfitting-and-overfitting.

[33] Arredondo-Santoyo, Marina, César Domínguez, Jónathan Heras, Eloy Mata, Vico Pascual, Mª Soledad Vázquez-Garcidueñas, and Gerardo Vázquez-Marrufo. “Automatic Characterisation of Dye Decolourisation in Fungal Strains Using Expert, Traditional, and Deep Features - Soft Computing.” SpringerLink. Springer Berlin Heidelberg, February 12, 2019. https://link.springer.com/article/10.1007/s00500-019-03832-8.

[34] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4510-4520. doi:10.1109/cvpr.2018.00474

[35]Niu, Qiya, Yunlai Teng, and Lin Chen. “Design of Gesture Recognition System Based on Deep Learning.” ResearchGate, February 2019. https://www.researchgate.net/publication/331675538_Design_of_gesture_recognition_system_based_on_Deep_Learning.

[36] P, Soumendra. “[Keras] a Thing You Should Know about Keras If You Plan to Train a Deep Learning Model on a Large...” Medium. fnp.dev, May 18, 2021. https://blog.fnp.dev/keras-a-thing-you-should-know-about-keras-if-you-plan-to-train-a-deep-learning-model-on-a-large-fdd63ce66bd2. 1

[37] “Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and All Those Confusing Names.” Github, May 23, 2018. https://gombru.github.io/2018/05/23/cross_entropy_loss/.

[38] Arora, Simrann. “The Inception Pre-Trained CNN Model.” OpenGenus IQ: Computing Expertise & Legacy. OpenGenus IQ: Computing Expertise & Legacy, May 28, 2020. https://iq.opengenus.org/inception-pre-trained-cnn-model/.

[39] Lin, Min, Qiang Chen, and Shuicheng Yan. “Network in Network.” arXiv.org, March 4, 2014. https://arxiv.org/abs/1312.4400.

[40] Buitinck, L., Louppe, G., Blondel, M., Pedregosa, Fabian, Mueller, A., Grisel, O., … Ga"el Varoquaux. (2013). API design for machine learning software: experiences from the scikit-learn project. In ECML PKDD Workshop: Languages for Data Mining and Machine Learning (pp. 108–122). https://scikit-learn.org/stable/modules/cross_validation.html

[41] Allibhai, Eijaz. “Holdout vs. Cross-Validation in Machine Learning.” Medium. Medium, October 3, 2018. https://medium.com/@eijaz/holdout-vs-cross-validation-in-machine-learning-7637112d3f8f.

Appendix

We show 10 images (Figure A1- Figure A8) of the architecture of the VGG16, VGG19, ResNet50, ResNet152, InceptionResNetV2, InceptionV3, MobileNet, MobileNetV2

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure A1: VGG16 ConvNet Configuration [13]

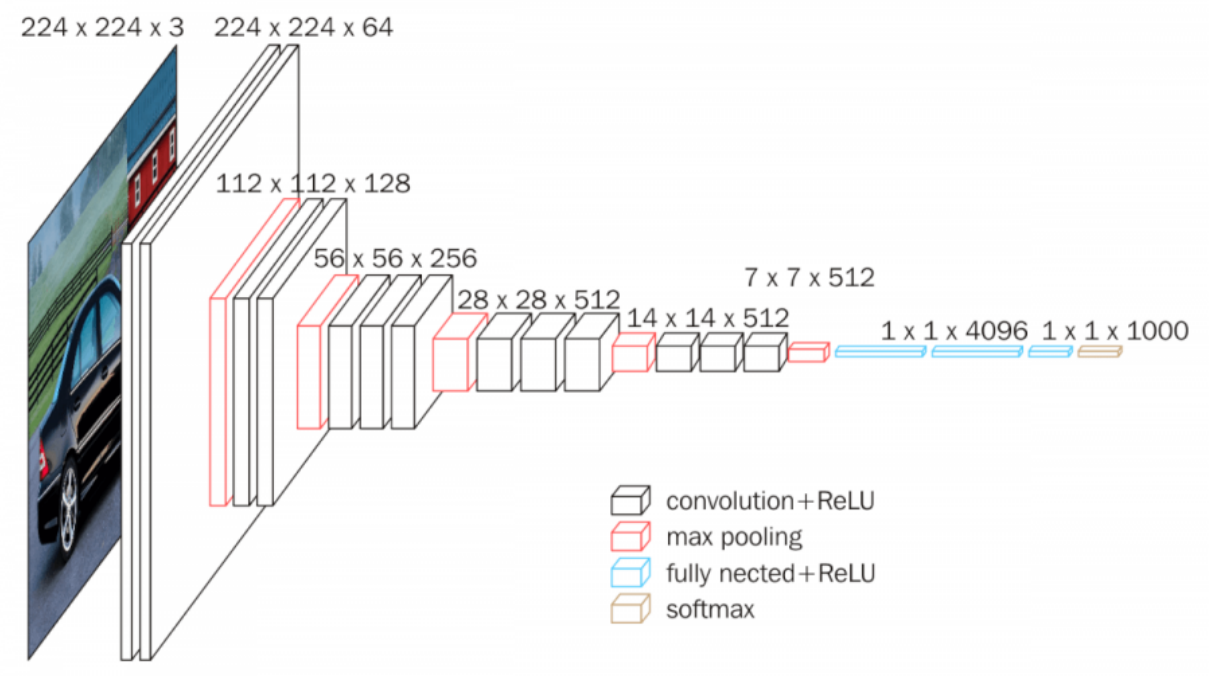


Figure A2. VGG16 Architecture [16]

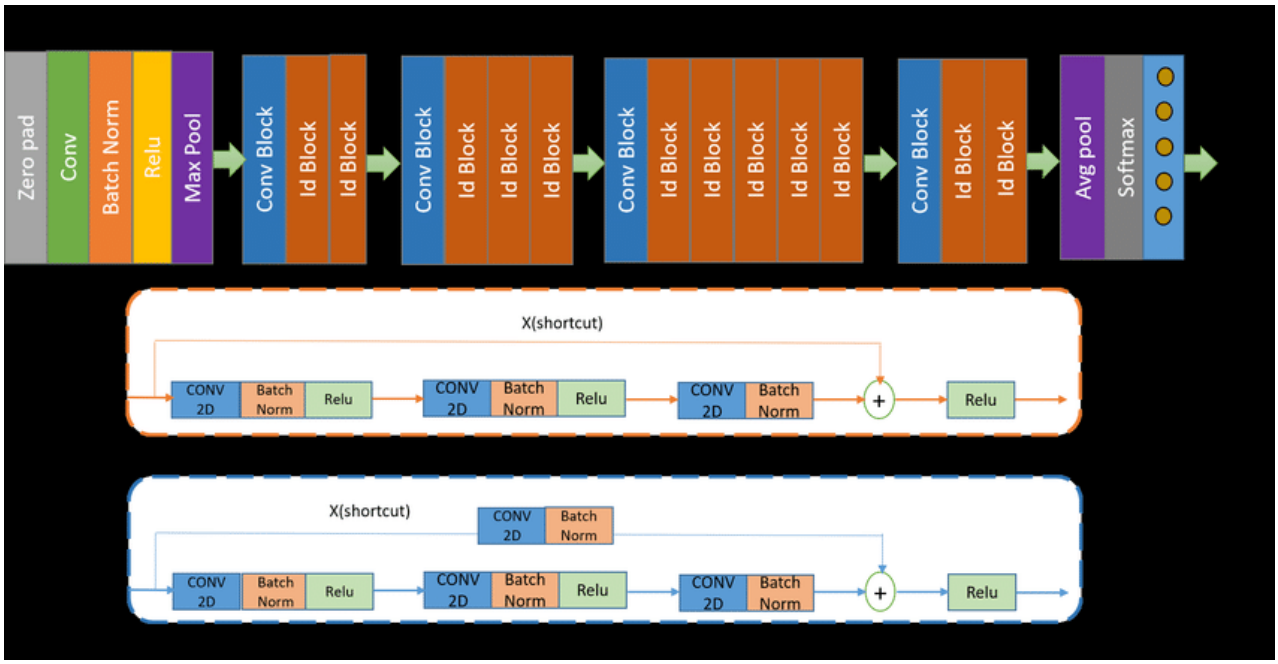


Figure A3: ResNet50 Architecture [17]

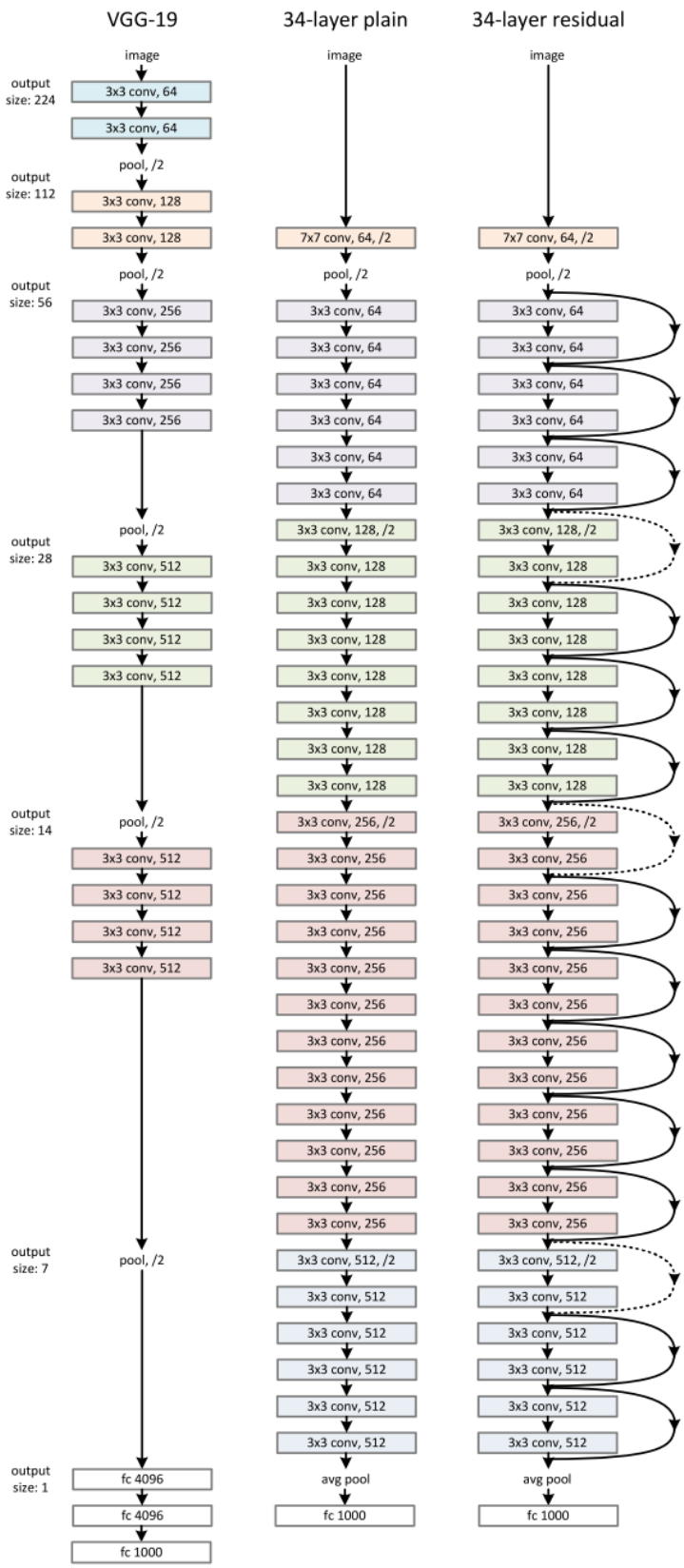


Figure A4. Resnet152 compared to VGG-19 [19]

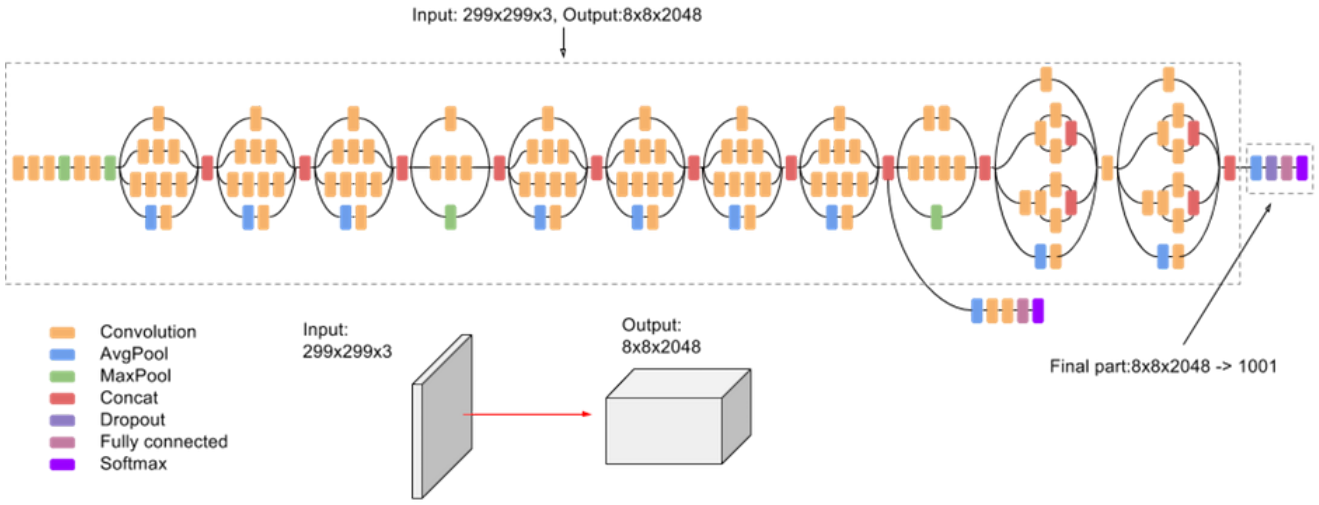


Figure A5: Inception-v3 basic architecture [21]

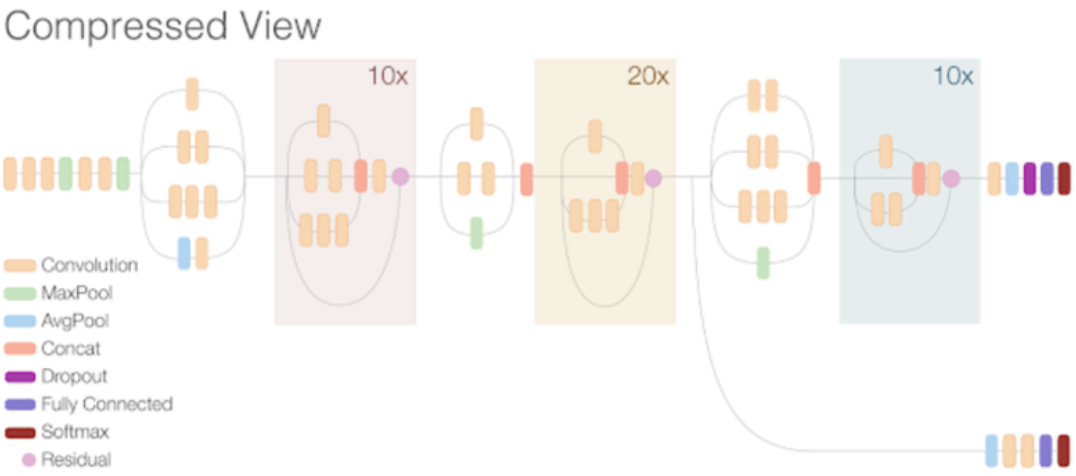


Figure A6: Inception-Resnet-v2 basic architecture [24]

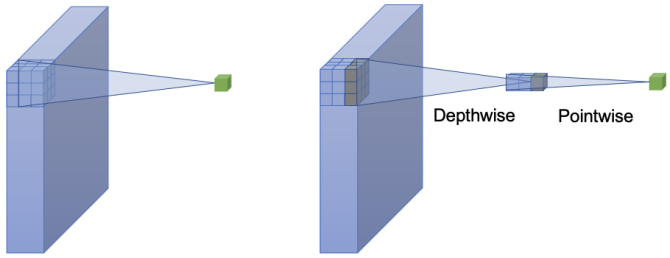


Figure A7: Standard convolution and depthwise separable convolution [https://arxiv.org/abs/1610.02357v3]

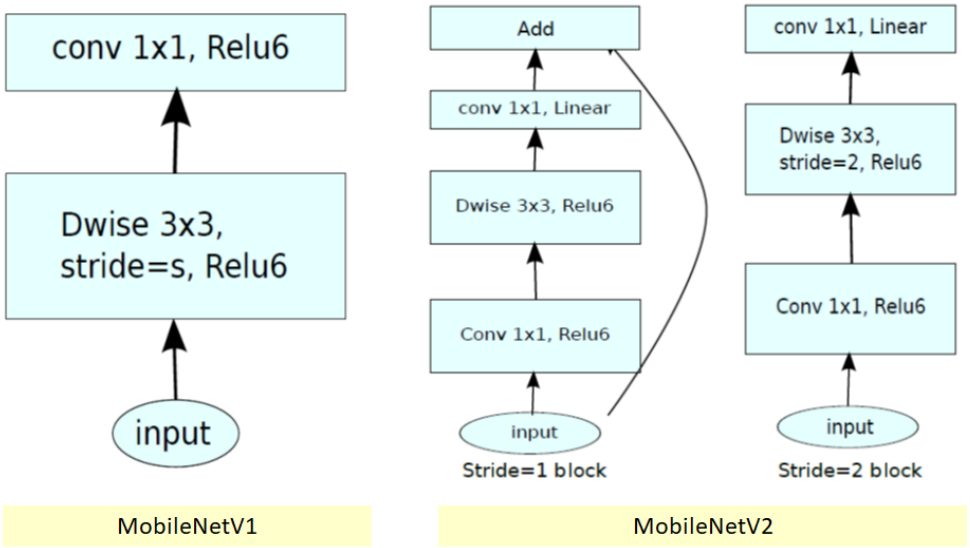


Figure A8: MobileNet & MobileNetV2 Differences [35]