A Comparative Study of Support Vector Machines and Convolutional Neural Networks for Object Detection Tasks Involving Satellite Imagery

Tudor Dorobantu
Tudor.Dorobantu@city.ac.uk

Abstract

This paper is motivated by the success of AI practices in generating Alternative Data strategies. Two Machine Learning algorithms are considered for a binary classification task involving detection of ships from satellite imagery. The models investigated are Support Vector Machine (SVM) and Convolutional Neural Networks (CNN). Model parameter choice are based on optimizing for accuracy scores while the F1 Score metric is used to compare the performance of competing algorithms. The Convolutional Neural Network proved to be slightly better in the context of the proposed classification task.

1. Brief Description and Motivation of the Problem

The investing community has come to rely more on quantitative methods to generate Alpha, a key performance indicator for funds and financial managers (Mullainathan, S. and Spiess, J., 2017). This reliance has led to new investment models that leverage cutting-edge advances in Machine Learning and Big Data. One new area in this field, named Alternative Data (strategies), leverages satellite imagery to understand economic phenomena underpinning the growth of markets. This trend has been fuelled by the ever-decreasing cost of sending satellites in space. An alternative data company named RS Metrics used geospatial images of parking lots of major US retailers to predict store visits and consequently revenue. RS Metrics alternative data investment strategy yielded a low correlation of 7.2% to the S&P 500 with a Sharpe ratio of 0.68 (JPMorgan Chase & Co., 2017). Other companies such as Orbital Insights use satellite imagery of petroleum tankers and storage facilities to predict the performance of commodities and oil & gas firms.

2. Summary of Studied Algorithms

2.1 Support Vector Machine (SVM)

In the context of a binary classification problem**,** Support Vector Machines seek to separate two classes of data points by drawing the optimal hyperplane between said classes (Cristianini, N. and Scholkopf, B., 2002). The process of finding the optimal hyperplane is computationally expensive and requires the data to be linearly separable. To overcome these issues, the SVM algorithm embeds the data in a higher dimensional space using the “kernel trick” (Cristianini, N. and Scholkopf, B., 2002). Thereafter, the optimal hyperplane is determined by the set of vectors of differing classes that are closest to each other in the new multi-dimensional space – these are known as support vectors. The optimal hyperplane for dividing the two classes will be the one that maximizes the margin between the hyperplane and the supporting vectors. In essence, the task of finding such a hyperplane reduces the problem to a quadratic programming paradigm (Cristianini, N. and Scholkopf, B., 2002). This is the key strength of the SVM algorithm since the objective function to be minimized is convex, thus removing the issue of local optimization that afflict deep neural network algorithms. The optimal solution for a classification problem is certain under polynomial time for SVMs (Cristianini, N. and Scholkopf, B., 2002).

Limitations of SVMs include long training times for large datasets and lack of classification probabilities.

2.2 Convolutional Neural Networks (CNN)

Convolutional Neural Networks are regarded as the most effective AI algorithms for pattern recognition in images, being designed to function similarly to the human visual cortex (O'Shea, K. and Nash, R., 2015).

CNNs extend the architecture of ANN by implementing a convolutional layer and pooling layer between the fully connected layer – these two types of layers have the role of reducing the number of neuron connections and thus the computational time required to train the network. The Convolutional layers convolute a set of filters over the original image in order to generate an activation map of lower dimensionality. The activation map usually contains certain image qualities such straight edges, text, and facial features. The pooling layer down samples the activation map further reducing the data dimensionality.

Limitations of CNNs include no guarantee of global minima and incomprehensible decision logic (Blackbox models).

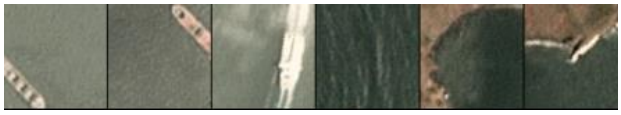
3. Dataset

3.1 Description of Dataset

The original dataset consists of 4,000 satellite images of format 80 x 80 RGB (Kaggle.com. 2018). The ship must be near-centered and fully captured in the image to be flagged as part of the positive class (ship). The presence of atmospheric conditions does not warrant exclusion from the positive class. There is a total of 1,000 positive examples in the dataset.

Figure 1- Sample of Positive Class Images

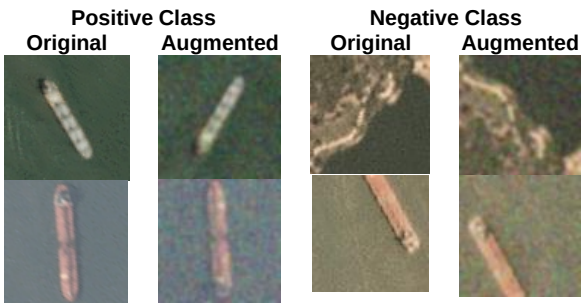
Figure 2- Sample of Negative Class Images



The other 3,000 satellite images encompass the negative class (no ship). A third of the images in the negative class are of landscapes (cities, sea, mountains and vegetation), another third of the examples contain images of partially framed ships and the last third contain images with geospatial features that have previously generated false positive with other machine learning models (Kaggle.com. 2018).

3.2 Dataset Augmentation

Figure 3 –Original vs. Augmented Images



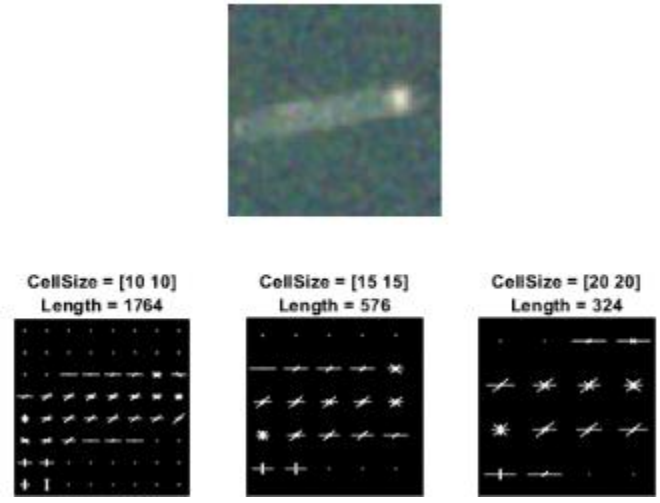
The entire image dataset has been augmented by flipping the images on their x and y axis and adding random noise and blur with a gaussian filter. Each of the original images in the positive class was copied three times over before being passed through the augmentation process. This was done to balance the dataset between positive and negative labeled instances. The tripling of the positive class does not result in similar samples since the image augmentation process entails stochastic procedures (e.g.

adding random noise, flipping on x and y axis). The augmented dataset does not contain the original images.

3.3 Exploratory Analysis

Identifying ships from satellite imagery can be regarded as an object detection task N. Dalal and B. Triggs (2006) propose the use of a Histogram of Oriented Gradients for detecting features sets for objection detection tasks. To successfully apply HOG for feature generation, the image pixels values are converted to greyscale and then binarized. The cell size parameter for the HOG transformation is deduced by visually inspecting a set of positive class samples across a range of cell sizes as can be seen in *Figure 4*.

Figure 4 – HOG Across Different Cell Sizes



The Cell Size of 15 x 15 was picked for use in training the SVM model as it is the parameter value that manages to best reduce data dimensionality while still preserving a pattern that discriminates the ship figure from its background.

4. Hypothesis Statement

Expectations of algorithm performance are based on the benchmark MNIST digit classification standings. Three of the top performing algorithms leverage Convolutional Neural Networks while SVM ranks in the middle of the leader board with a classification error twice as high as the best performing CNN-based algorithm (MNIST on Benchmarks.AI, 2020).

As such, we hypothesis that CNN model will outperform SVM when it comes to accuracy scores. Nonetheless, we expect that leveraging HOG method for feature generation for the SVM model will minimize the performance gap between the two algorithms. This belief is motivated by the performance improvements in classification accuracy of SVM with HOG highlighted in N. Dalal and B. Triggs (2006) study.

5. Methodology

In order to yield more robust models that are not prone to overfitting, training and testing is performed on an augmented data set that adds noise, blur, vertical and horizontal flips to the original images. The image augmentation also produces a balanced data set that assures accuracy scores will not be affected from models overfitting to the majority class (Shorten, C. and Khoshgoftaar, T.M., 2019).

Random sampling is used to generate a test set containing 10% of all images. The rest 90% of images are reserved for training and validating the two competing models. Hyperparameter selection is based on reducing 10-fold cross validation error (*CV Error*). After the validation phase all models are trained on the entire validation set.

Data pre-processing is performed for both models. For the SVM model a 15 x 15 Cell Size HOG feature set is generated. Images are transformed to grayscale for the CNN model.

F1 scores are used to draw a conclusion on algorithm choice at the testing phase.

6. Choice of Parameters and Experimental Results

6.1 Support Vector Machine (SVM)

A two-phase Bayesian Optimization process was used whereby the Kernel Function and Box Constraint are first optimized and then passed as inputs for the second optimization phase which seeks to maximize the 10-fold cross validation score for the Kernel Scale and Polynomial Order parameters.

The results of the optimization process are presented below.

Table 1 – Optimization Results for SVM

Model	Kernel Function	Box Constraint	Kernel Scale	Poly. Order	10-fold CV Error
Baseline	Gaussian	1	1	N/A	94.17%
Optimal	Polynomial	0.0040	0.7407	4	98.18%

Figure 5

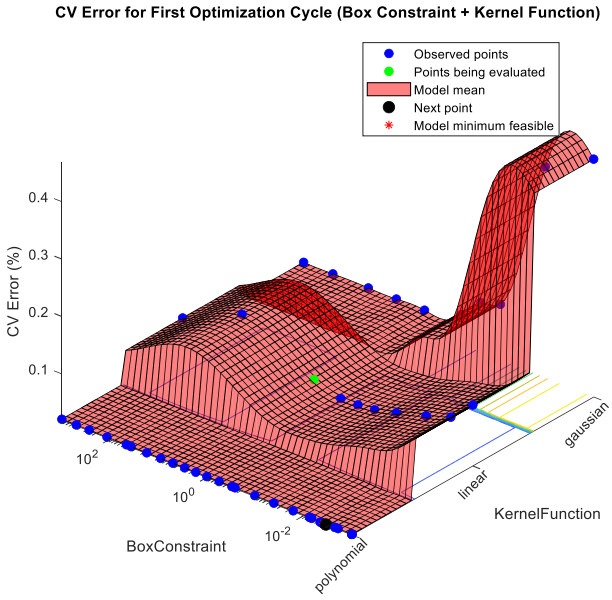
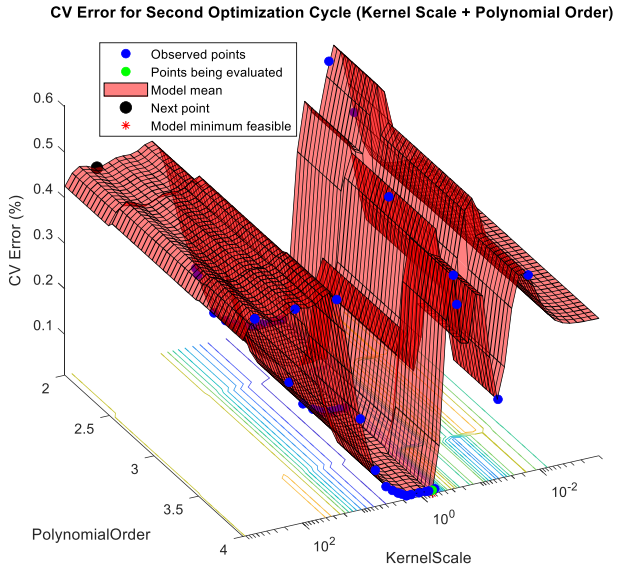


Figure 6



6.2 Convolutional Neural Networks (CNN)

A two-phase Random Grid Search was used whereby the CNN network architecture is first optimized. Architecture parameters that are optimized are filter numbers of each convolution layer, pooling value and activation function. A small learning rate of 1e-4 is used at this stage to assure efficient convergences. The optimal architecture is then used to find the best Solver Equation, Learning Rate and Momentum/Gradient Decay parameters.

The results of the Optimization process are presented below.

Table 2 – CNN Opt. Results for Architecture Values

Rank	Filter Numbers*	Pooling Value	Activation Function	10-fold CV Error
BEST PERFORMING VALUES				
#1	**[32 | 16 | 8]**	**2 X 2**	**Leaky ReLU**	**99.96%**
#2	[32 | 16 | 8]	2 X 2	ReLU	99.91%
#3	[32 | 16 | 8]	5 X 5	Leaky ReLU	99.87%
#4	[32 | 16 | 8]	4 X 4	Leaky ReLU	99.76%
LEAST PERFORMING VALUES				
#1	[2 | 2 | 2]	3 X 3	Leaky ReLU	94.19%
#2	[2 | 2 | 2]	5 X 5	ReLU	95.39%
#3	[4 | 2 | 2]	4 X 4	ReLU	95.74%
#4	[2 | 2 | 2]	4 X 4	Leaky ReLUI	96.19%

Table 3 – CNN Opt. Results for Solver Eq., Learn Rate and Mom./Grad. Decay

Rank	Solver Equation	Learning Rate	Momentum/ Gradient Decay**	10-fold CV Error
BEST PERFORMING VALUES				
#1	**sgdm**	**1e-3**	**0.9**	**100%**
#2	sgdm	1e-2	0.9	100%
#3	sgdm	1e-2	0.3	100%
#4	rmsprop	1e-4	0.9	100%
LEAST PERFORMING VALUES				
#1	adam	1e-2	0.9	98.61%
#2	rmsprop	1e-2	0.999	98.72%
#3	rmsprop	1e-2	0.9	99.37%
#4	adam	1e-2	0.99	99.67%

**Legend for notation* - [filter # at first conv. layer | filter # at second conv. layer | filter # at third conv. layer]
**For sgdm solver the parameter is Momentum while for rmsprop and adam the hyperparameter is Gradient Decay.

Table 4 – Final Optimization Results for CNN

Model	Filter Numbers*	Pooling Value	Activation Function	Solver Eq.	Learning Rate	Momentum	10-fold CV Error
Baseline	[8 | 4 | 2]	2 X 2	ReLU	sgdm	1e-5	0.9	71.89%
Optimal	[32 | 16 | 8]	2 X 2	Leaky ReLU	sgdm	1e-3	0.9	100%

**Legend for notation* - [filter # at first conv. layer | filter # at second conv. layer | filter # at third conv. layer]

7. Analysis and Critical Evaluation of Results

Table 5 – Testing Performance Results

Model	Accuracy	Recall	Precision	F1 Score
SVM	98.33%	98.67%	98.01%	98.34%
CNN	98.33%	99.33%	97.39%	98.35%

SVM proved be almost as effective as the Convolutional Neural Network Model. The SVM model tended to have a higher Recall error while the CNN yielded a larger Precision Error.

7.1 Support Vector Machine (SVM)

In the first phase of the optimization process, the polynomial kernel function vastly outperformed all other kernel functions across the entire Box Constraint spectrum (*see Figure 5)*. In the second optimization phase (*see Figure 6),* the kernel scale parameter had a substantial effect on reducing the accuracy error while the polynomial order parameter only slightly reduced the error.

Overall, Bayesian Optimization yielded parameters that are in line with those suggest in Scientific Literature (Cortes, C. and Vapnik, V., 1995) – e.g. Optimal SVM Models tend to have close to zero Kernel Scale and Box Constraint parameter values.

7.2 Convolutional Neural Networks (CNN) – *(see Table 2 & 3 for Parameters)*

In the case of the Convolutional Neural Network, accuracy was substantially sensible to the number of filters. The largest value for the number of filters parameter ([32 16 8]) was present in the top four best performing algorithms while the lowest filter parameter led to the top four worst performing algorithms. This was to be expected as a larger number of filters per convolutional layer allows the algorithm to pick more of the images features and thus generate better activation maps(O'Shea, K. and Nash, R., 2015).

The best performing models also had the lowest pooling value parameter. Again, this is to be expected as a large pooling value leads to down sampling of the activation maps.

The only difference between the best and second-best performing algorithm was the activation function. A Leaked ReLU function yielded a performance improvement of 0.05%. This suggest that the CNN may be experiencing inactive (dead) neurons. Nonetheless, more data is required to reach such a conclusion since the worst performing algorithms paint an inconclusive picture in regard to the activation function parameter.

8. Conclusion and Future Work

The SVM model performed surprisingly well for the image classification task at hand. The success of the SVM model should be attribute to the effective usage of HOG for featuring mapping rather than a failure to optimize the CNN layers. HOG feature generation step acted as the convolutional layer in the CNN, reducing data dimensionality while accurately discriminating key features in the image.

An interesting area for further study would be to leverage the power of Autoencoders in order to remove noise and create lower dimensional representations of features in images that then can be fed into an SVM model.

9. References

Benchmarks.ai. 2020. MNIST On Benchmarks.AI. [online] Available at: <https://benchmarks.ai/mnist> [Accessed 1 April 2020].

Cortes, C. and Vapnik, V., 1995. Support-vector networks. Machine learning, 20(3), pp.273-297.

Cristianini, N. and Scholkopf, B., 2002. Support vector machines and kernel methods: the new generation of learning machines. *Ai Magazine*, *23*(3), pp.31-31.

Hastie, T., Friedman, J. and Tibshirani, R. (2017). The Elements of statistical Learning. 2nd ed. New York: Springer, pp.417-438.

JPMorgan Chase & Co., 2017. *Machine Learning And Alternative Data Approach To Investing*. Big Data and AI Strategies. [online] JPMorgan Chase & Co. Available at: <https://faculty.sites.uci.edu/pjorion/files/2018/05/JPM-2017-MachineLearningInvestments.pdf> [Accessed 1 April 2020].

Kaggle.com. 2018. Ships In Satellite Imagery. [online] Available at: <https://www.kaggle.com/rhammell/ships-in-satellite-imagery> [Accessed 1 April 2020].

Mullainathan, S. and Spiess, J., 2017. Machine learning: an applied econometric approach. *Journal of Economic Perspectives*, *31*(2), pp.87-106.

N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection", Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol. 1, pp. 886-893, 2005.

O'Shea, K. and Nash, R., 2015. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

Shorten, C. and Khoshgoftaar, T.M., 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1), p.60.

10.Glossary

Optimal CNN Complete Architecture

```
15x1 Layer array with layers:

    1   'imageinput'    Image Input             80x80x1 images with 'zerocenter' normalization
    2   'conv_1'        Convolution             32 3x3x1 convolutions with stride [1  1] and padding 'same'
    3   'batchnorm_1'   Batch Normalization     Batch normalization with 32 channels
    4   'leakyrelu_1'   Leaky ReLU              Leaky ReLU with scale 0.01
    5   'maxpool_1'     Max Pooling             2x2 max pooling with stride [2  2] and padding [0  0  0  0]
    6   'conv_2'        Convolution             16 3x3x32 convolutions with stride [1  1] and padding 'same'
    7   'batchnorm_2'   Batch Normalization     Batch normalization with 16 channels
    8   'leakyrelu_2'   Leaky ReLU              Leaky ReLU with scale 0.01
    9   'maxpool_2'     Max Pooling             2x2 max pooling with stride [2  2] and padding [0  0  0  0]
   10   'conv_3'        Convolution             8 3x3x16 convolutions with stride [1  1] and padding 'same'
   11   'batchnorm_3'   Batch Normalization     Batch normalization with 8 channels
   12   'leakyrelu_3'   Leaky ReLU              Leaky ReLU with scale 0.01
   13   'fc'            Fully Connected         2 fully connected layer
   14   'softmax'       Softmax                 softmax
   15   'classoutput'   Classification Output   crossentropyex with classes 'negative' and 'positive'
```

Overview of Deep Learning Model Architecture

Deep Learning algorithms such as CNNs map features to labels by leveraging the distributed and parallel computational power of the Directed Acyclic Graph structure with the linear classification algorithm of the perceptron. A typical neural network has an input layer, *n* hidden layers and an output layer.

In a supervised learning context involving the binary classification of images, each pixel value of the image (the feature set) is fed to a corresponding node in the input layer. The values in the input layer lead to the activation of a set of neurons in the subsequent layer and so on. At the end of the network there are only two neurons representing the corresponding labels. (SOURCE) If the network is trained well, the neuron representing the true label should activate. If the outcome is incorrect, the weights and biases that govern the activation of individual neurons are adjusted by the backpropagation algorithm during the learning process.

Limitations of ANNs in the Context of Image Classification

In an Artificial Neural Network (ANN), each neuron is connected to all neurons in the subsequent layer. This becomes an issue when dealing with images as the number of weights that need to be optimized grows with the expansive feature set of the image data. Training ANNs for image classification problems becomes computationally expensive, as such CNNs where developed to deal with the high dimensionality of image data.

Alpha

Key performance indicator for financial investments. Alpha is the difference in returns of an investing strategy from the market benchmarks. A financial manages main goal is to increase Alpha

Sharpe Ratio

The measure of risk-adjusted return for a financial investment.

11.Implementation Details

- To run trained models only please run runModels_script.m.
- To download the original data please go to the link in the LINK_TO_ORIGINAL_DATA.txt. Save the data in the augmented_data folder for the Experiment Livescript to be able to access it. Alternatively, the data can be downloaded by following the link here: https://www.kaggle.com/rhammell/ships-in-satellite-imagery
- To run the entire experiment pipeline please run runExperiments_livescript.mlx. Make sure that the data is in the augmented_data folder. The livescript will function with both the augmented and original data set without requiring any modifications.
- To augment the image data (e.g. add noise, gaussian blur, etc.) please use the ImageAugment.py class. Instructions on how to use the class should be available on the github page -> https://github.com/tdorobantu/ImageAugment