Ships Identification in Satellite Images

Instructor: Professor Peter Gerstoft **TA:** Mark Wagner, Paolo Gabriel, Nima Mirzaee

Team members: Weilun Zhang , Zhaoliang Zheng, Mingchen Mao

SCRIPPS INSTITUTION OF OCEANOGRAPHY UC San Diego

Predicting

There is a need for machine learning algorithms to help automate the satellite image analysis process which can be applied to issues such as monitoring port activity and supply chain analysis。

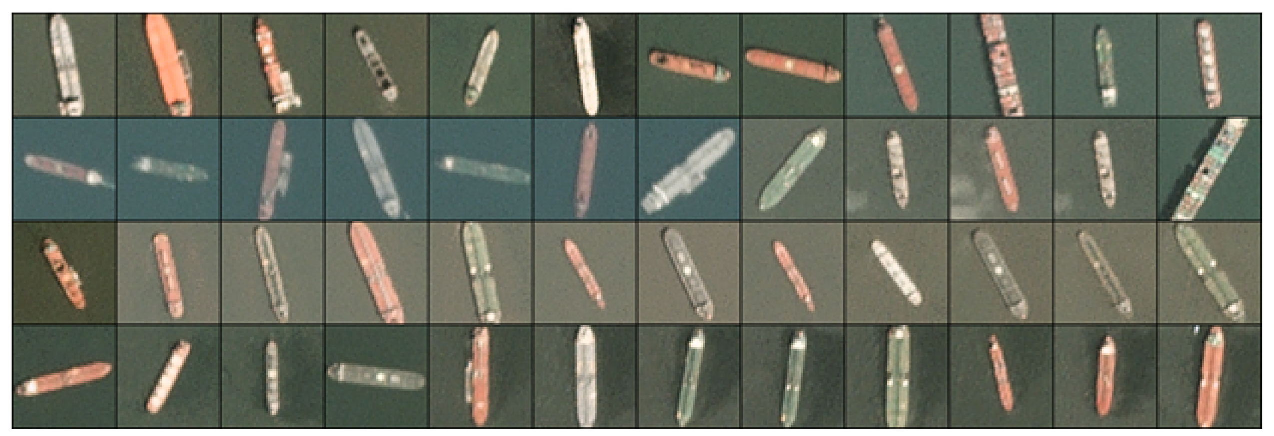
By taking advantage of our training data sets, we expected to develop an algorithm that is able to identify ships in a given satellite image with at least 90% accuracy. Different models were implemented and a multi-model model with various weight distribution might be applied when if necessary.

Data

Our data mainly fall into 2 categories:

- Cropped satellite image with labels, used for model training
- Raw & real satellite image, used for training examination

The cropped satellite images were preprocessed into 3 color channels (RGB), which were normalized manually in our code.

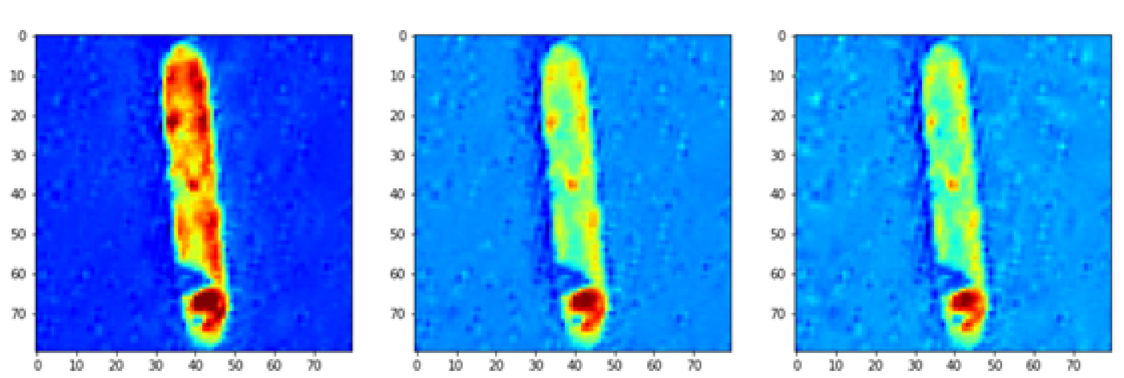


Feature

The main features of our raw input data are colors and their corresponding locations.

Data preprocessing

Reshape data
Normalize data
Shuffle all indexes
Output encoding



Reference:

[1] T.N. Arnesen, R.B. Olsen, D.J. Weydahl, "Ship detection signatures in AP Mode data", 56th International Astronautical Congress, pp. 06, 2005.

[2] Krogager, E.; Heiselberg, H.; Møller, J.G.; von Platen, S. Fusion of SAR and EO imagery for Arctic surveillance. In Proceedings of the NATO IST-SET-128 Specialist Meeting, Norfolk, VA, USA, 4–5 May 20.

[3] Daniel, B.; Schaum, A.; Allman, E.; Leathers, R.; Downes, T. Automatic ship detection from commercial multispectral satellite imagery. Proc. SPIE 8743 2013.

[4] Gade, M.; Hühnerfuss, H.; Korenowski, G. Marine Surface Films; Springer: Heidelberg, Germany, 2006.

Models

Try multiple tree models first to find the best top models:

(1) LogisticRegression
(2) SVC & LinearSVC
(3) XGBClassifier
(4) RandomForestClassifier
(5) DecisionTreeClassifier
(6) KNeighborsClassifier
(7) GaussianNB
(8) GradientBoostingClassifer
(9) Convolution neural network + SGD optimizer

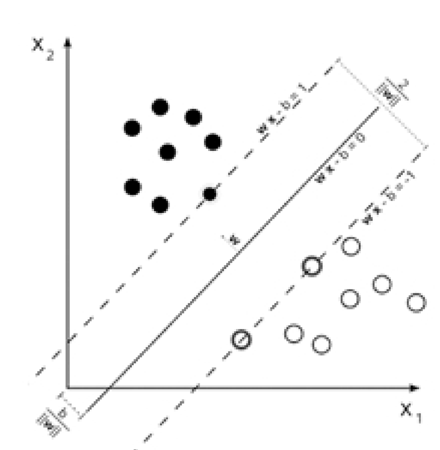
```
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.svm import LinearSVC
from xgboost import XGBClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import GradientBoostingClassifier
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.optimizers import SGD
```

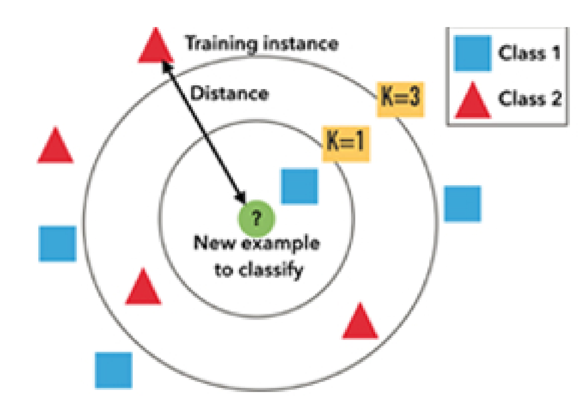
Models introduction

(1) Logistic Regression (LR):

$$log(p) = b_0 + b_1X_1 + b_2X_2 + \ldots + b_kX_k$$

(2) Support vector machineSVM \ LSVM:





(3) EXtreme Gradient Boosting (XGB):

Gradient Boosting + Stochastic Gradient Boosting + Regularized Gradient Boosting

(4) Gradient Boosting Classifier (GBC):

$$F_m(x) = F_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} \mathbf{1}_{R_{jm}}(x),$$

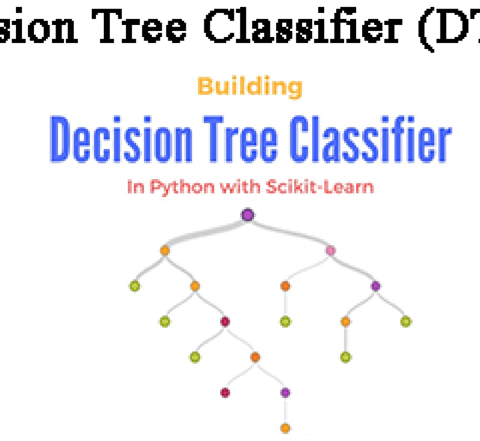
$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, F_{m-1}(x_i) + \gamma)$$

(5) Random Forest (RF):

$$C_{bag}(x) = \arg\max_m \{C(S^m, x)\}_{m=1}^{M}$$

$$C_{bag}(x) = \arg\max_m \{C(S^m, x)\}_{m=1}^{M}$$

(6) Decision Tree Classifier (DTC):



(7) KNeighborsClassifier(KNN):

It runs through the whole dataset computing d between the given point x and each training observation according to Euclidean distance. We' ll call the K points in the training data that are closest to x the set S. It then estimates the conditional probability for each class. Finally, our input x gets assigned to the class with the largest probability.

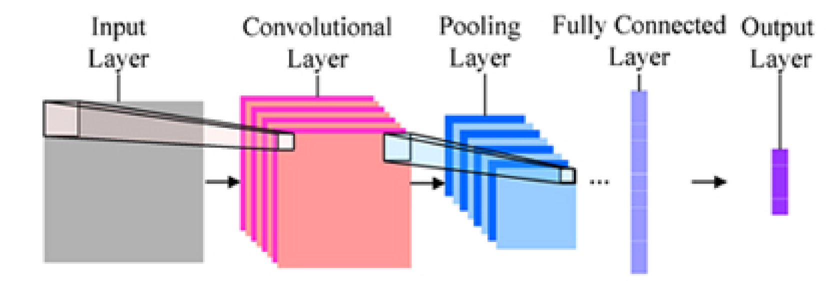
$$d(x, x') = \sqrt{(x_1 - x_1')^2 + (x_2 - x_2')^2 + \ldots + (x_n - x_n')^2}$$

$$P(y = j | X = x) = \frac{1}{K} \sum_{i \in A} I(y^{(i)} = j)$$

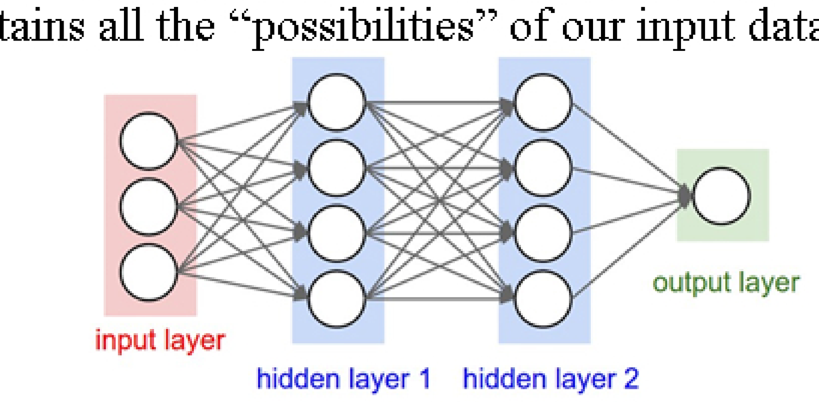
(8) Naive Bayes Classifier (NBC):

$$p(x = v | C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{\frac{-(v-\mu_k)^2}{2\sigma^2_k}}$$

(9) Convolutional Neural Network (CNN) :



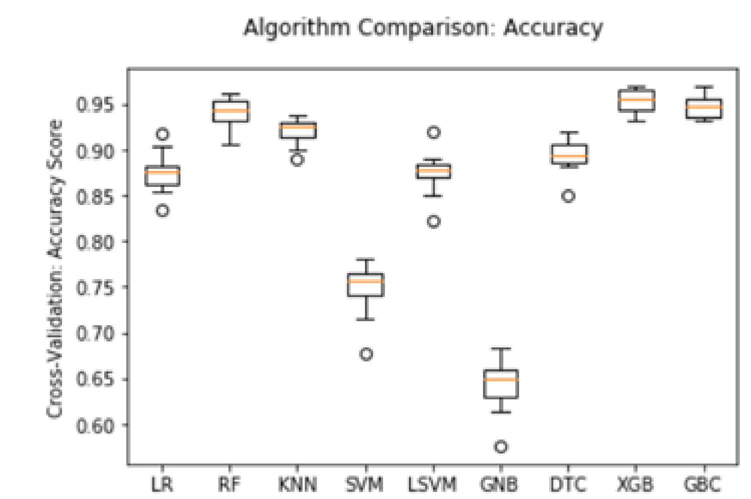
Convolutional Neutral Network (CNN) is a fairly popular and important algorithm used in machine learning. The basic idea of CNN is to apply a convolution operation to the input and pass the convoluted input, which is often referred as a layer, to the next layer. By repeating this process, we can build a multi-layered convolution network, which contains all the "possibilities" of our input data.

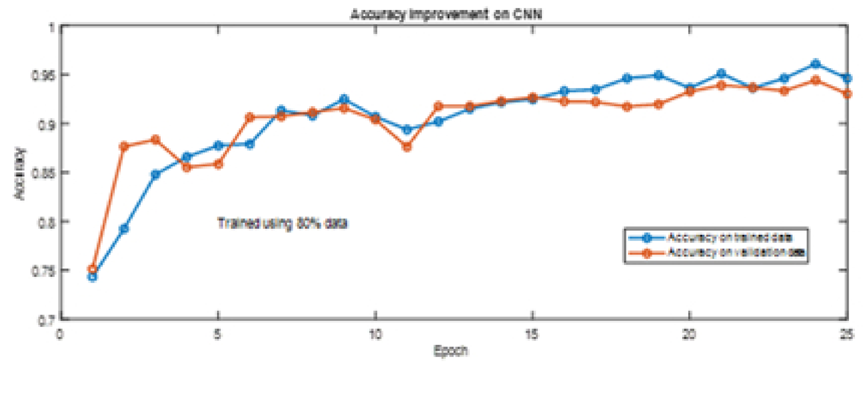


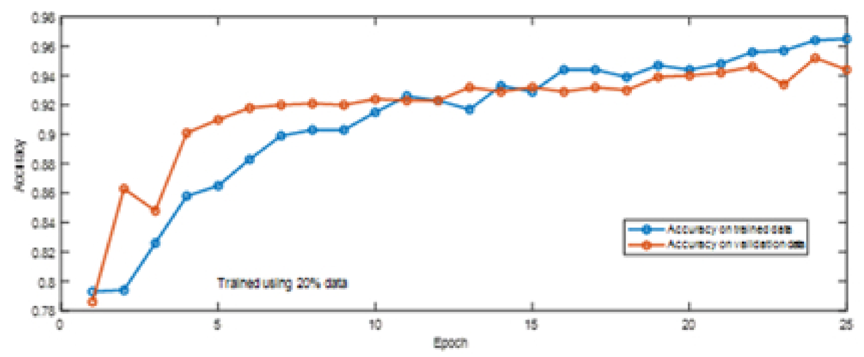
(10) Stochastic Gradient Descent (SGD):

$$w := w - \eta \nabla Q(w) = w - \eta \sum_{i=1}^{n} \nabla Q_i(w)/n.$$

Result Plot:



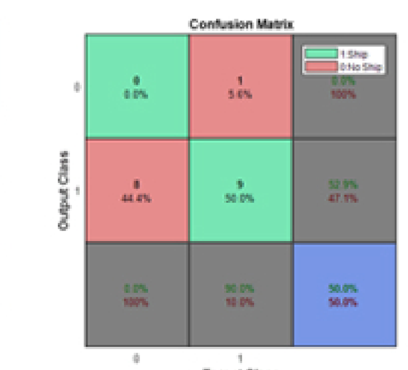





```
LR = LogisticRegression
RF = RandomForestClassifier
KNN = KNeighborsClassifier
SVM = Support Vector Machine SVC
LSVM = LinearSVC
GNB = GaussianNB
DTC = DecisionTreeClassifier
XGB = XGBClassifier
GBC = GradientBoostingClassifier
```

Result Discussion:

	Mean of Accuracy	Standard Deviation of Accuracy
GNB	0.642014	0.028610
SVM	0.746875	0.028986
LR	0.874653	0.022312
LSVM	0.875000	0.024006
DTC	0.893056	0.018228
KNN	0.919792	0.014768
RF	0.940625	0.016757
GBC	0.947222	0.012306
XGB	0.953472	0.011948
CNN with SGD	0.966725	0.009720



It turns out that each less data to train yields better results than training using more data. One potential explanation is that the key features of our model is compact and obvious, using more data tends to increase the correlation between these key features, which hampers the ability of our algorithm to distinguish between ships and no-ships.

Identify ships in satellite images

Future steps:

- Apply more preprocessing techniques on the real satellite images, such as separate the land and the ocean which allows us to only identify ships on the ocean.
- Use better tuning methods to tune hyperparameters on multi-parameter models such as XGBoost when powerful computing resource is available.
- Combining different models with different weight distributions to maximize the benefits of each model.