Prediction of FOD type on an airplanes runway using Supervised Machine Learning Techniques

Ertugrul Aypek
Department of
Computer Engineering
Middle East Technical University
2171270
Email: ertugrul.aypek@metu.edu.tr

Rashad Gasimov
Department of
Computer Engineering
Middle East Technical University
2351179
Email: e235117@metu.edu.tr

Elif Ozturk
Department of
Computer Engineering
Middle East Technical University
2350965
Email: elifozturk@posteo.net

***Abstract*—In this study, Foreign Object Debris types have been classified based on image data. Support Vector Machines (SVM), Decision Trees and k-Nearest-Neighbors have been chosen as learning algorithms. This research shows, that CNN can be used as just a feature extractor and that other machine learning models can replace CNN's classification task. By doing so, depending on the model, performance can be increased.**

I. Introduction

With the increasing development in technology, also the usage of airplanes increases. In fact, airplanes are the number one choice for travelling between countries. Science and Engineering make it possible to build robust machines, that will not crash due to every unexpected situation. Even though these machines are built like that, accidents occur frequently. One possible cause for an airplane accident is when there are foreign objects on the runway that could damage the motors. Therefore even small objects are dangerous. A known scenario in which a small object caused a tragic accident is the crash of Air France Flight 4590 in 2000, when a metal object fell off of another engine on the runway and shattered one of 4590s tires, causing the death of 113 people. To prevent accidents like that, those foreign objects need to be detected and their type need to be classified properly, so that further solutions can be developed to make a safer runway. For this reason this project aims to focus on different classification algorithms to predict those FOD-types (Foreign Object Debris), which can be metal, plastic or a bird.

II. Background Information

As the objective is to prevent airplane engine from unexpected threats like undetected objects and items, main target is to detect the type of object and predict its potential pitfalls. Object type recognition is a challenging problem because the shape of objects belonging to different classes is not well defined. Features and predictions computed from image classification algorithms and selection of classifier plays an important role in this class detection problem. For every of three labels, the dataset will be divided in a way that there will be same amount of data for each class, so there will be no class imbalance problem. Finally, dividing datasets into various categories will eventually lead determining concerned FOD-types.

III. Literature Review

As autonomy industry rises, there is more interest for image classification. There are many algorithms/methods in the literature to solve image classification problem. Some of the main methods are Support Vector Machine, Artificial Neural Networks, Decision Trees, K-Nearest Neighbor and Random Forest as covered in [1] and [3]. Another method is Convolutional Neural Networks which is a subset of Artificial Neural Networks and currently one of the most popular methods to classify images [2]. One can choose one of these algorithms or a combination of these algorithms with the consideration of accuracy-speed trade-off for a specific purpose. The choice of an algorithm for a specific problem is done via trial and error approach. Figure 1, shows the results (in terms of accuracy) of experiments done in [3] for binary classifications of 11 different classes and indicates best model for each class. These can be deduced from Figure 2 that there is no best universal algorithm and performances of each algorithm might change on different problems. However, it can also be deduced that there are some robust algorithms which are likely to take over the others.

Fig. 1: Image containing a number of FOD instances

MODEL	CAL	COVT	ADULT	LTR.P1	LTR.P2	MEDIS	SLAC	HS	MG	CALHOUS	COD	BACT	MEAN
BST-DT	PLT	**.938**	.857	**.959**	**.976**	.700	.869	**.933**	.855	**.974**	**.915**	.878*	**.896***
RF	PLT	.876	.930	.897	.941	**.810**	.907*	.884	.883	.937	.903*	.847	.892
BAG-DT	–	.878	.944*	.883	.911	.762	.898*	.856	**.898**	.948	.856	**.926**	.887*
BST-DT	ISO	.922*	.865	.901*	.969	.692*	.878	.927	.845	.965	.912*	.861	.885*
RF	–	.876	.946*	.883	.922	.785	.912*	.871	.891*	.941	.874	.824	.884
BAG-DT	PLT	.873	.931	.877	.920	.752	.885	.863	.884	.944	.865	.912*	.882
RF	ISO	.865	.934	.851	.935	.767*	**.920**	.877	.876	.933	.897*	.821	.880
BAG-DT	ISO	.867	.933	.840	.915	.749	.897	.856	.884	.940	.859	.907*	.877
SVM	PLT	.765	.886	.936	.962	.733	.866	.913*	.816	.897	.900*	.807	.862
ANN	–	.764	.884	.913	.901	.791*	.881	.932*	.859	.923	.667	.882	.854
SVM	ISO	.758	.882	.899	.954	.693*	.878	.907	.827	.897	.900*	.778	.852
ANN	PLT	.766	.872	.898	.894	.775	.871	.929*	.846	.919	.665	.871	.846
ANN	ISO	.767	.882	.821	.891	.785*	.895	.926*	.841	.915	.672	.862	.842
BST-DT	–	.874	.842	.875	.913	.523	.807	.860	.785	.933	.835	.858	.828
KNN	PLT	.819	.785	.920	.937	.626	.777	.803	.844	.827	.774	.855	.815
KNN	–	.807	.780	.912	.936	.598	.800	.801	.853	.827	.748	.852	.810
KNN	ISO	.814	.784	.879	.935	.633	.791	.794	.832	.824	.777	.833	.809
BST-STMP	PLT	.644	**.949**	.767	.688	.723	.806	.800	.862	.923	.622	.915*	.791
SVM	–	.696	.819	.731	.860	.600	.859	.788	.776	.833	.864	.763	.781
BST-STMP	ISO	.639	.941	.700	.681	.711	.807	.793	.862	.912	.632	.902*	.780
BST-STMP	–	.605	.865	.540	.615	.624	.779	.683	.799	.817	.581	.906*	.710
DT	ISO	.671	.869	.729	.760	.424	.777	.622	.815	.832	.415	.884	.709
DT	–	.652	.872	.723	.763	.449	.769	.609	.829	.831	.389	.899*	.708
DT	PLT	.661	.863	.734	.756	.416	.779	.607	.822	.826	.407	.890*	.706
LR	–	.625	.886	.195	.448	.777*	.852	.675	.849	.838	.647	.905*	.700
LR	ISO	.616	.881	.229	.440	.763*	.834	.659	.827	.833	.636	.889*	.692
LR	PLT	.610	.870	.185	.446	.738	.835	.667	.823	.832	.633	.895	.685
NB	ISO	.574	.904	.674	.557	.709	.724	.205	.687	.758	.633	.770	.654
NB	PLT	.572	.892	.648	.561	.694	.732	.213	.690	.755	.632	.756	.650
NB	–	.552	.843	.534	.556	.011	.714	-.654	.655	.759	.636	.688	.481

Fig. 2: Results of different algorithms' binary classifications on 11 classes [3]

IV. Methodology

In these days, CNN is one of the most popular methods to solve image classification problem. However, our aim is to do image classification with different approaches and to compare those, to be able to decide the best suitable algorithm for this dataset. Our approach is to take a combination of CNN and one algorithm out of Support Vector Machines (SVM), Decision Trees and k-Nearest-Neighbor.

Feature extraction is an important part in machine learning. CNN uses an automatic feature extractor without user interaction developed by Google [6]. It is shown by Horn et. al. (2017) that using CNN features might have a boosting effect on performance of machine learning models [5]. After a CNN training session is finished, it is possible to get automatically extracted features which is called transfer learning [4]. In this work, to get reliable features, CNN will be used to extract the features. After features are automatically extracted, they will be fed into chosen machine learning algorithms to create our model.

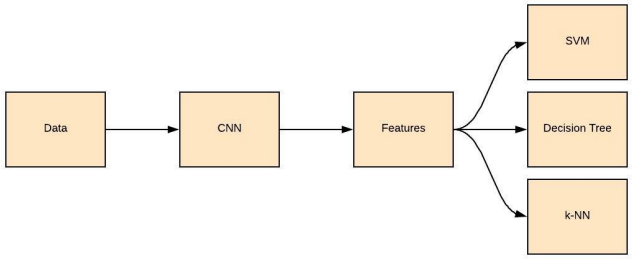

Fig. 3: A graph containing workflow

V. Evaluation

In the field of machine learning, feature extraction and selection had always been a problematic issue. Typically features are extracted manually through different approaches, such as converting image data to time series or simply using the pixel values. Not only extracting features, but also choosing the most informative ones is a time consuming task. However, since deep learning algorithms have been grown up, this task has become easier. One of those deep

learning algorithms is CNN, which is usually used to classify images. As stated in [7], using CNN features might advance performance of classification results. The main goal of this project is combining feature extraction and selection parts of CNN with classical machine learning algorithms. To achieve this, a CNN model has been trained and cut at it's dense layer. The outputs at that layer were then used as the desired features. Those features were fed into classical machine learning algorithms, such as Decision Trees, Support Vector Machines and k-Nearest-Neighbors. A visualized version of workflow can be found in Figure 3. These algorithms have been trained with different hyper-parameters and their results have been compared.

In this project, to evaluate all of the resulted models, hold-out method was used. In the datasets, there were 1000 training and 144 test images for each class. All of the data consist of labeled images. The input image size is 200x200. Before feeding the data into CNN, data preprocessing on raw images has already been done. Data preprocessing involves converting RGB images to grayscale and normalizing those grayscale images to fit into the interval of [0,1]. Doing so decreases the influence of noises and outliers.

A. Decision Tree

Decision Trees are well-known top-down, non-linear and non-parametric classifiers. They are made up of edges and nodes. Following some if-else branches, an input ends up with a leaf node. Leaf nodes are outputs i.e. class labels in classification problem. Although decision trees are non-parametric, there are some limitations in order to get best performance results on test data. These limitations are chosen as split criterion, maximum depth and minimum samples on a leaf node. So, in this project, decision trees are trained and tested on four attributes which are features, criterion, maximum depth of tree and minimum samples on a leaf node.

There are two feature types which are CNN and classical. CNN indicates that the features fed into Decision Tree model are extracted and selected by CNN model. Classical indicates that the features are extracted by classical machine learning methods. Although CNN extracts common image features like color, shape, texture or histogram of gradients; it is still a black box that exactly what the model is learning. Despite this mystery, it is a good learner. So, in this research, as an expected result, CNN features gave much more high performance results since it extracted more informative features than classical methods.

There are two criterias which are gini and entropy. These criterias are functions to measure the quality of each split in decision tree. According to this quality calculated for each feature that is fed into the model, features with highest quality are chosen as top nodes' split decider. Gini is interpreted as the gini impurity and entropy is interpreted as the information gain. Their mathematical formulas are given as:

TABLE I: Results of Decision Tree method

Features	Criterion	Depth	Min Samples	F1	Precision	Accuracy
CNN	gini	6	10	.906	.907	.905
CNN	gini	6	20	.905	.905	.905
CNN	gini	6	50	.872	.874	.873
CNN	gini	7	10	.898	.899	.898
CNN	gini	7	20	.903	.903	.903
CNN	gini	7	50	.872	.874	.873
CNN	entropy	6	10	.898	.898	.898
CNN	entropy	6	20	.900	.901	.900
CNN	entropy	6	50	.894	.894	.893
CNN	entropy	7	10	.905	.908	.905
CNN	entropy	7	20	.903	.905	.903
CNN	entropy	7	50	.893	.894	.894
classic	gini	6	10	.687	.709	.682
classic	gini	6	20	.677	.696	.673
classic	gini	6	50	.664	.685	.662
classic	gini	7	10	.702	.730	.699
classic	gini	7	20	.689	.712	.687
classic	gini	7	50	.669	.698	.668
classic	entropy	6	10	.670	.705	.671
classic	entropy	6	20	.673	.703	.673
classic	entropy	6	50	.629	.654	.629
classic	entropy	7	10	.696	.708	.694
classic	entropy	7	20	.688	.690	.687
classic	entropy	7	50	.654	.658	.652

$$Entropy = -\sum_{i=1}^{n} p_i log p_i \quad (1)$$

$$Gini = 1 - \sum_{i=1}^{n} p_i^2 \quad (2)$$

There are two maximum depth of tree which are chosen as 6 and 7. Maximum depth has a great influence on decision tree models. As the model gets deeper and deeper so the training accuracy gets higher i.e overfitting on training data. In this project, these numbers for maximum depth are chosen as not to be too big or small in order to avoid underfitting or overfitting on the training data.

Minimum number of samples on a leaf has also a big importance on decision tree models. If it is too small, then there may be some splits for just a small portion of training data to be classified correctly. However, if it is too big, then there may not a learning process. In this project it has been chosen as 10, 20 and 50.

The results in terms of F1 score, precision score and accuracy score can be found in TABLE I for different configurations of parameters. An example visualized tree with the configuration (CNN features, gini criterion, 6 max depth, 50 min samples) can be found at this address.

B. Support-Vector-Machine

SVM is a supervised learning algorithm, that separates the data linearly with the highest possible gap between each class. The separation is done by constructing hyperplanes. Those hyperplanes should have the highest possible margin to the nearest datapoints (support vectors). In cases where the data is not linearly separable, kernel functions can be used. That allows increasing the space, where the hyperplane

is constructed to a higher dimension, where the data gets linearly separable. In this project, linear SVMs were used, as well as nonlinear classifiers by using the polynomial and radial basis kernel function. The following kernel functions were used:

linear: $k(x, y) = x^T y + c$

polynomial: $k(x, y) = (\alpha x^T y + c)^d$

radial basis function: $k(x, y) = exp(-\gamma||x - y||^2)$

Where c is an optional constant, α is the slope parameter, d is the degree of the polynomial and γ is the hyper-parameter that needs to be tuned.

To get the most informative results the model with both methods, classical feature extraction and feature extraction with CNN, was trained with different C and Gamma values. In SVM the C-hyper-parameter is used as a regularization parameter. This parameter allows avoiding overfitting by constructing a less complex function. The lower C, the larger can the margin and the simpler can the function get. This is demonstrated in TABLE II: By choosing a smaller C ($C = .1$), the accuracy gets less, which could be caused by overfitting. Gamma shows the influence of a single datapoint. For too small Gammas, the model is less likely to overfit, but it is more likely to get a less appropriate boundary that captures the shape of the data correctly, which means it will lead to a higher bias and lower variance. For too large Gammas, it is more likely to overfit, because the influence of the support vectors will get less and no regularization can be done. On the other hand, a larger Gamma can capture the shape of the data better, hence a larger Gamma will lead to a lower bias and higher variance. In TABLE II, it is visible that by taking a smaller ($\gamma = 0.001$) and a larger ($\gamma = 0.006$) value, the overall accuracy gets less, which means it either overfits or is too simple. Whereas, taking Gamma as $\gamma = 1/n\_features = 1/512 = 0.002$ gets the best overall accuracy.

Furthermore, the accuracies of classical feature extraction and CNN's feature extraction were compared. It is clearly visible from TABLE II that extracting features with a CNN model gets better results than with the classical method, which is due to the fact that more informative features are chosen in CNN. Also, tuning C and γ does not change the fact that classically extracted features lead to poorer accuracies.

C. k-Nearest-Neighbors

k-NN algorithm is one of the simplest and most used classification algorithm. It is a lazy classifier since it does not do training. For k-Nearest-Neighbors algorithm, the main idea is finding k closest neighbors and classifying them in order to find a proper class for the target point. The algorithm then combines the labels of those training datapoints to determine the testing datapoint's label. For our k-Nearest-Neighbors classifier, Euclidean distance has been used as the distance measurement method. A model for each configuration has been trained separately and accuracy, precision and f-score have been measured for it. Referring to TABLE III, it can be observed that the performance of the k-Nearest-Neighbor classifier changes regarding the type of features, value of k modifies. In general, for the CNN model, the accuracy of predictions, precision and F1 scores are higher than classical models since CNN extracts more precise and useful features for image classification. Interestingly, $k = 100$ Nearest Classifier models performs better in all configurations comparing to $k = 200$ models.

TABLE II: Results of SVM method

Features	C	Gamma	Kernel	F1	Precision	Accuracy
CNN	1.0	.002	rbf	.937	.938	.937
CNN	1.0	.002	linear	.95	.951	.95
CNN	1.0	.002	poly	.965	.95	.95
CNN	.1	.002	rbf	.867	.871	.87
CNN	.1	.002	linear	.95	.95	.95
CNN	.1	.002	poly	.883	.888	.883
CNN	1.0	.001	rbf	.91	.913	.91
CNN	1.0	.001	linear	.95	.951	.95
CNN	1.0	.001	poly	.883	.888	.883
CNN	.1	.001	rbf	.883	.887	.883
CNN	.1	.001	linear	.95	.95	.95
CNN	.1	.001	poly	.772	.805	.777
CNN	1.0	.006	rbf	.897	.912	.897
CNN	1.0	.006	linear	.95	.951	.95
CNN	1.0	.006	poly	.909	.91	.91
CNN	.1	.006	rbf	.65	.8	.67
CNN	.1	.006	linear	.95	.95	.95
CNN	.1	.006	poly	.95	.95	.95
classic	1.0	.002	rbf	.693	.693	.655
classic	1.0	.002	linear	.667	.687	.667
classic	1.0	.002	poly	.668	.667	.667
classic	.1	.002	rbf	.61	.625	.613
classic	.1	.002	linear	.667	.669	.67
classic	.1	.002	poly	.683	.57	.669
classic	1.0	.001	rbf	.63	.667	.627
classic	1.0	.001	linear	.65	.705	.667
classic	1.0	.001	poly	.603	.642	.6
classic	.1	.001	rbf	.603	.641	.594
classic	.1	.001	linear	.67	.704	.667
classic	.1	.001	poly	.492	.559	.494
classic	1.0	.006	rbf	.617	.674	.614
classic	1.0	.006	linear	.67	.705	.667
classic	1.0	.006	poly	.629	.664	.627
classic	.1	.006	rbf	.369	.554	.387
classic	.1	.006	linear	.67	.704	.667
classic	.1	.006	poly	.67	.739	.667

TABLE III: Results of k-NN method

Features	Number of Neighbors	F1	Precision	Accuracy
CNN	100	.891	.892	.892
CNN	200	.866	.868	.866
Classic	100	.596	.696	.604
Classic	200	.575	.721	.588

VI. Conclusion

Main purpose of this research was to compare different machine learning algorithms and also different feature extraction methods on FOD images. The hypothesis was that extracting features with CNN and feeding these features into machine learning algorithms would lead to better results. As it can be deduced from all experimental results for each algorithm,

this hypothesis is verified. No matter which hyper-parameter configuration is chosen, classical feature extraction always led to poorer accuracies. Additionally, among all machine learning algorithms used, SVM resulted in the best performance and k-NN resulted in the worst performance. As a much more important outcome, SVM's best accuracy (0.95) was higher than the CNN model's accuracy (0.94) which was used to extract features. As a future work, different layers of CNN model can be used as a feature extractor instead of the dense layer.

REFERENCES

[1] P. P. Naswale, P. E. Ajmire. Image Classification Techniques- A Survey. International Journal of Emerging Trends Technology in Computer Science, Vol. 5(2), 2278-6856, (2016).

[2] Y. LeCun, P. Haffner, L. Bottou, Y. Bengio. Object Recognition with Gradient Based Learning, (1999).

[3] R. Caruanna, A. Niculescu-Mizil. An Empirical Comparison of Supervised Learning Algorithms, (2006).

[4] M. Farooq, E. Sazonov. Feature Extraction Using Deep Learning for Food Type Recognition, (2017).

[5] Z.C. Horn, L. Auret, J.T. McCoy, C. Aldrich, B. M. Herbst. Performance of Convolutional Neural Networks for Feature Extraction in Froth Flotation Sensing, (2017).

[6] Abadi et. al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems, (2015).

[7] R. Paul et. al. Deep Feature Transfer Learning in Combination with Traditional Features Predicts Survival Among Patients with Lung Adenocarcinoma, (2016).