AIMLRCS: A MACHINE LEARNING APPROACH TO SPACECRAFT ATTITUDE AND OBJECT IDENTIFICATION BASED ON RCS FROM THE S3TSR

Carlos Paulete[(1)], David Cano[(2)], Jan Siminski[(2)], Cristina Pérez[(3)], Diego Escobar[(1)], Jesús Tirado[(1)]

[(1)] GMV, Isaac Newton 11, P.T.M. Tres Cantos, Madrid, Spain. Email: cpauleteperianez@gmv.com, descobar@gmv.com, jtirado@gmv.com
[(2)] ESA/ESOC, Robert-Bosch-Str. 5, 64293 Darmstadt, Germany, Email: david.cano.mananes@esa.int, jan.siminski@esa.int
[(3)]CDTI, Cid 4, Madrid Spain. Email: cristina.perez@cdti.es

ABSTRACT

One of the natural by-products of the radar measurements is the Radar Cross Section (RCS). It is given as a sequence in time and is a measure of the radar reflectivity of an object from the point of view of the radar installation. There is plenty of information about an object, implicitly contained within the RCS sequence it generates. However, that information is somehow embedded and it is not straightforward to extract. The evolution of the RCS differences contains information on the radar scattering characteristics of different viewing perspectives on the spacecraft. The characterization of these signals relies on the non-isotropic nature of the radar scattering from an object with respect to its attitude towards the line of sight of the radar.

The RCS sequences are exploited in two ways within the context of this activity. Firstly, the attitude stability mode of the Resident Space Object (RSO) is extracted to classify whether an object is stable, tumbling, etc. Then, based on the same information, a more fine-grained classification algorithm is developed, being able to associate each RCS sequence with the corresponding object from a corpus of known spacecraft. In both cases, a Machine Learning (ML) algorithm is responsible for the characterization, with additional advanced pre-processing steps that increase the performance of the ML model.

The combination of both classifiers allows for the continuous tracking of the evolution of the attitude stability of a set of spacecraft using only passively obtained information from the radar installation. That is, given an RCS sequence, the knowledge of the owner RSO and its current and past attitude stability mode allows for a close following of the evolution of the stability of objects.

There is a great value in the usage of the RCS for this application, as it is the minimum output of a radar installation. Currently object identification based on radar measurements relies on correlation of the orbital track detected by the radar and a library of continuously updated orbital tracks. This application would reduce the reliance on said track library and provide independent way of whitelisting the objects directly by the radar installation, with the added benefit of providing simultaneously the attitude stability of the identified object.

Different ML algorithms are trained with real data obtained with the Surveillance Radar (S3TSR), which is part of the Spanish Space Surveillance and Tracking (S3T) programme. The obtained results are systematically compared in this study to measure the accuracy of the ML algorithms.

1 INTRODUCTION

Radar installations all over the world continuously generate RCS sequences from bodies in orbit overhead. Within the context of this activity, an approach is proposed to exploit them to produce additional information about the observed objects.

Most RSOs have anisotropic radar refraction properties that are unique for each object. However, the correlation between RCS sequence and an object characteristic is not straight forward. Therefore, the use of ML algorithms is proposed in this study to identify that relationship between the RCS evolutions and the associated objects.

ML algorithms provide a framework for a general map function between an input (e.g. the RCS sequence and metadata) and an output (e.g. the attitude stability of the RSO) if enough data is available for the training process.

In cooperation with ESA and CDTI, past observations made on known objects from the S3TSR installation are provided for use in this activity. This dataset allows for complex ML models to be trained which are used for two tasks:

- Attitude stability classification
- Direct object identification

The first one is further broken down in two: 4-class and 3-class attitude stability classification. The former classifying RSOs in Earth pointing, inertia pointing, geodetic and tumbling objects and the latter is an easier case where Earth and inertia pointing objects are aggregated under a single stable category. In any case, the model is only allowed to use RCS sequences and the metadata generated alongside it to make the classification.

2 STATE OF THE ART

The RCS is a natural product of any radar installation. As such, there is widespread interest in the research community to explore ways to extract value out of the RCS sequences produced during standard operations. From these sequences, the research field is split in two main applications, the determination of the attitude stability of the spacecraft or the direct identification of the object.

2.1 Attitude Stability Classification

The research on the extraction of the attitude stability is mainly concerned with determining if the body is stable in space and, if this is not the case, obtaining the tumbling frequency of the object of interest. This is possible because most spacecrafts have a non-isotropic radar signature, that is, they provide hugely different RCS signals depending on the attitude they present to the radar line of sight.

An object can be classified as three-axis stabilised by analysing the randomness of its RCS sequence. During a pass, if the satellite is stable, the variations of its RCS signal are small and have a random behaviour [1], [2]. If this check fails and the object is determined not stable, a frequency analysis can be carried out to determine its tumbling period. In the literature, this is done via an autocorrelation function analysis [2] or a Cepstrum analysis [1]. Furthermore, from the period extracted the object can be classified as tumbling or spin-stabilised since tumbling objects typically have a much lower rotation frequency.

There are other, more complex, methods in the literature that try to directly obtain the attitude of the spacecraft and its angular velocity [3], [4]. To achieve this, a RCS simulator is created, to simulate an RCS signal. Then, a loss function is defined that compares the RCS sequence of the target with one generated with guesses on the orientation and angular velocity. Finally, a particle swarm optimization algorithm sets out to find the attitude of the simulation that best matches the target RCS sequence. Particle swarm optimization is an optimization algorithm similar to gradient descent but using a set of points instead of only one initial point. These points travel down the local gradient but, additionally, they influence the path of the others if they are in a region in the space with a better loss function. This algorithm, although more costly than gradient descent, allows for better exploration of the solution space such that it is less likely to miss the global optimum.

2.2 Direct Object Identification

Direct object identification aims at classifying specific spacecraft amongst a set of them based on the idea that every space object has a different radar signature over the time that can be differentiated from the rest of the population under study. This can be done on a limited corpus of objects by using as features an estimated object size as well as other statistical features of the RCS sequence considered. All in all, this particular application is more complex than attitude stability classification as it involves classifying signals into as many classes as objects in the library.

In the reviewed literature, identification efforts are focused on differentiating between a small library of objects, from differentiating between a RCS sequence belongs or not to only one specific object [5] to up to 13 different spacecraft [9]. The first approach could be used in a one vs many classifier that can be trained for each specific object in a larger dataset of bodies.

The sequences are never fed directly into the actual classifier algorithm. Instead, some feature extraction steps are used first. Typically, some statistical features are used like the mean, standard deviation, median and so on [8], [9]. On top of that, Wavelet Transform (WT) (a similar transformation to the FFT) is used to extract a matrix that is then distilled into a set of features to be fed to the classifier [10], [5]. Instead of using the WT in reference [9], the authors opt for using an estimated effective diameter based on the Size Estimation Model developed by NASA [11].

These features are fed into a classifier that identifies the spacecraft that generated them. In most cases, the classification model chosen is a Nearest Neighbour Fuzzy Classifier [8], [9]. Less often, the model is a Set-Valued Method [5], and in only one case a Neural Network is used [10]. In this last reference, the authors use a set of features coming from a WT and feed it to a feedforward Neural Network (NN) of three layers of 25 neurons each. The classification task between 5 objects was very successful using very low computational resources. This approach is very promising as it aligns with the objectives of the activity significantly even if attitude stability classification is considered.

3 DATASET DESCRIPTION

This section is devoted to the discussion of the general properties of the dataset and the implications these characteristics have on a classifier trained with it. The analysis will cover size properties of the dataset as well as the labelling processed followed to assign attitude stability mode.

The dataset used contains 10818 individual Tracking Data Messages (TDMs). A TDM contains a series of observations with additional metadata about how the measurement was taken. Several features are provided for each observation, but, for this application, the most interesting ones are: RCS, observation epoch, azimuth and elevation. The dataset contains TDMs that span from the 01/07/2019 until 30/09/2020. Each one is labelled with the cospar number of the object correlated with the observations made.

Each TDM contains a minimum of 4 observations and a maximum of 173. A more in depth look at the temporal characteristics of the TDMs within the dataset is presented in Figure 3 1. It shows the distributions of total TDM duration time, number of observations per TDM and mean time between observations in each TDM. The histograms show how the distributions of the presented variables vary wildly. Consider the two rightmost plots in Figure 3-1, they are related with the frequency resolution that could be extracted from the RCS sequence. The time between observations limits the maximum observable frequency to≈2 Hz. This frequency may not be enough to capture the effects of spinning objects with traditional Fourier transform.

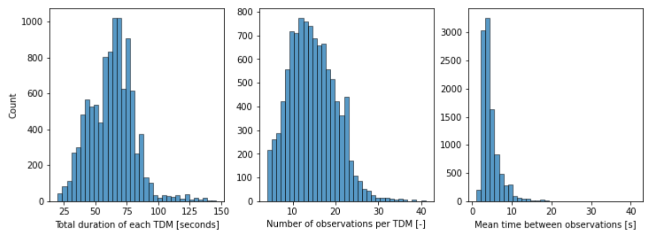


Figure 3-1. Temporal features of the TDM dataset.

As the TDMs are not labelled with the attitude stability mode of the observed RSO, they were subsequently assigned based on the object's mission. The initial labelling is done as follows:

- Earth pointing: Active objects in LEO whose main mission can be gathered under Earth observation.
- Inertia pointing: Active objects in LEO whose main mission is scientific but is not Earth observation. These are mainly telescopes which look at celestial objects for long periods of time.
- Geodetic spheres: Radar calibration spheres in LEO.
- Tumbling: Old disabled missions in LEO and available rocket bodies.

From the point of view of attitude classification, the dataset is distributed as shown in Figure 3-2. As is apparent from the figure, the dataset is highly unbalanced. Most TDMs come from Earth pointing objects, closely followed by tumbling objects and, finally, with very low counts, inertia pointing and geodetic objects appear. This points at a problem that must be dealt with when training a model. If there are many more samples of one class over another, a classifier will learn to be biased towards the more populous class, such that its overall accuracy increases. However, this will yield lower performance on the less populous class. This problem will be revisited when actually describing training methods.

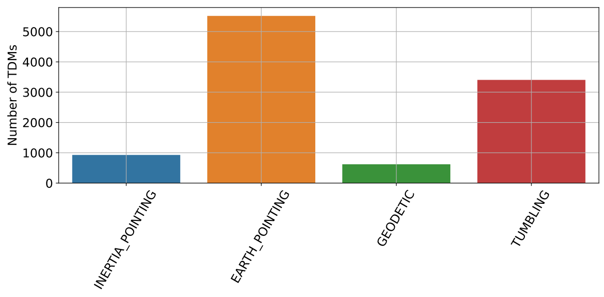


Figure 3-2. Number of TDMs in each attitude stability class

From the point of view of object identification, 46 individual objects are considered and its distribution is shown in Figure 3-3. There are large disparities between the numbers of TDMs associated to each object. It ranges from the smallest set at 69 TDMs for JASON_1 to the largest at 614 for ARIANE_40_RB. This may pose a problem in classification as biases toward more frequent objects may appear.

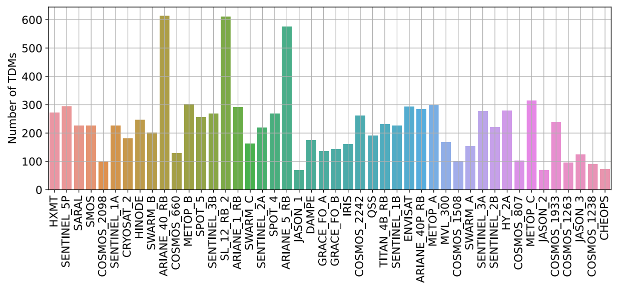


Figure 3-3. Number of TDMs for each RSO.

3.1 RCS Signal Biases

The RCS signal produced by an RSO is dependant mainly on two things: Geometry and surface properties. Since most objects in the database use completely different platforms, the distribution of the RCS values observed over time is more or less unique. However, for the purposes of attitude stability classification, it is interesting to ensure that the distribution along each of the 4 classes considered (Earth pointing, inertia pointing, geodetic and tumbling) is similar. This is needed because the model may learn to distinguish objects based on the strength of the signal alone instead of the characteristics of the RCS sequence. However, looking at Figure 3-4, it is clear that this is not the case in the dataset.

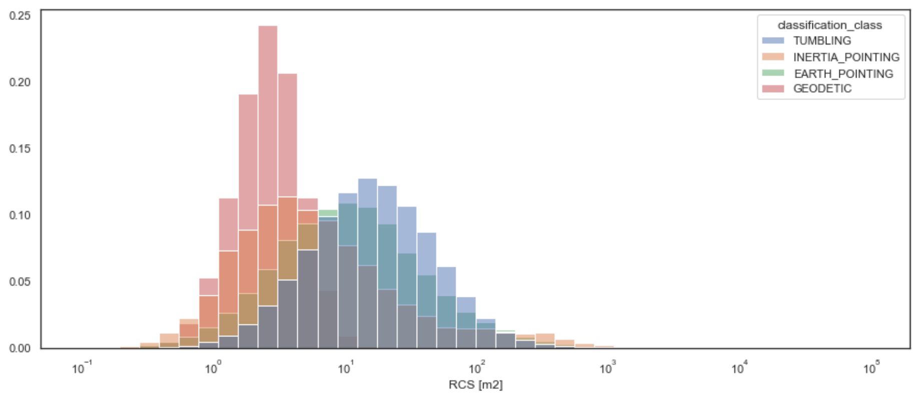


Figure 3-4 Distribution of RCS values by attitude stability population

Tumbling objects, for instance, tend to be larger than the others. This happens because most known tumbling objects are rocket bodies which are physically larger than other operational RSOs. To alleviate this problem, for the case of attitude stability classification, the RCS values will be normalized such that they all have a minimum 0 value and a maximum of 1. This will not be done in the case of object identification where physical size may be a determining factor.

3.2 Additional features

In order to increase the overall performance of the models, several features were created from the ones already available. Four more features were created per observation. They are rotations of vector that goes from the sensor to the observed RSO.

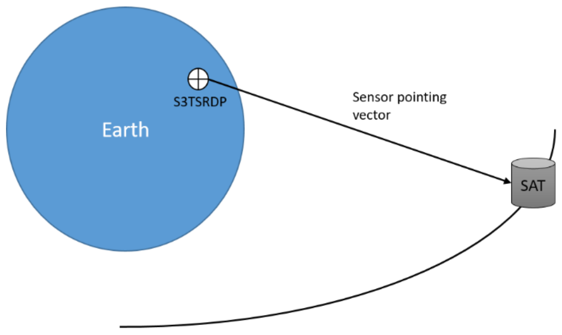


Figure 3-5. Schematic of the sensor point of view vector.

The four features are the azimuth and elevation of the sensor point of view in two different reference frames: ITRF [12] and GCRF [13].

ITRF is an Earth centred Earth fixed reference frame. The idea behind using it is that it preserves a point of view on Earth pointing objects that is fixed whereas the sensor frame is not. GCRF on the other hand is Earth centred as well but inertial. It is supposed to do the same job but for inertia pointing objects in this case.

ITRF is an Earth centred Earth fixed reference frame. The idea behind using it is that it preserves a point of view on Earth pointing objects that is fixed whereas the sensor frame is not. GCRF on the other hand is Earth centred as well but inertial. It is supposed to do the same job but for inertia pointing objects in this case.

4 METHODOLOGY

This section is devoted to the discussion of the model architecture used for the classifiers as well as the methods used to train such models. This section is generic over the classifier type, that is, the same overall architecture is used for all attitude and object classifiers diverging only in the target of the classification.

4.1 Model Architecture

On a high level, the model is composed of a Long Short Term Memory (LSTM) layer that feeds into a feedforward layer that, in turn, makes the prediction through a sigmoid activation function. The complete architecture is shown in Figure 4-1

As shown in the figure, each observation contained in one or several TDMs is fed into the LSTM recurrent layer. The output of this recurrent layer feeds into the feed forward block though a dropout layer. Then, the data flows through a single hidden layer with Rectified Linear Unit (ReLu) activation function. Finally, the output of this layer goes through batch normalization, another dropout and the output layer which has a log-softmax activation layer. In the following paragraphs each of these terms will be explained in detail.

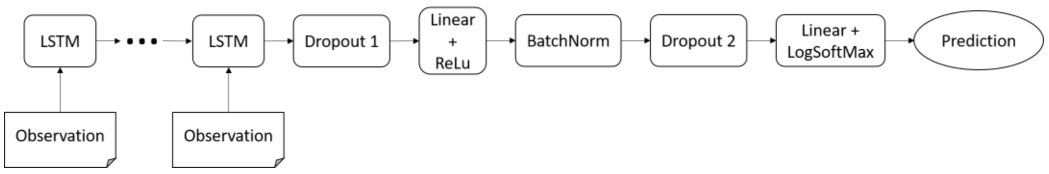


Figure 4-1Model architecture

The LSTM or Long Short Term Memory cell is a type of recurrent cell designed to avoid the vanishing gradient problem caused by ordinary Recurrent Neural Networks (RNNs) over long sequences. It does so by introducing a memory cell. It can carry information over long sequences with gates that allow more information in or discard some of the cell. LSTM networks also solve the problem of having instances in the training set with a different number of inputs (i.e. different number of observations). Thus, there is no need to sample the observations in each TDM to guarantee having all the instances with the same sequence length. The feature vector then may be better renamed as a feature matrix because it is 2D. Then, the shape of this feature matrix is $N_{observations}$ by $N_{features}$. For example, in a sample with 3 observations and using as features RCS, azimuth and elevation from the sensor, the shape would be 3 by 3.

A linear cell is simply a linear combination of the inputs plus a bias value. The activation function on the other hand is critical to the performance of the model. In this case two different functions are used. ReLu or Rectified Linear Unit is very simply a function $y = x$ when $x > 0$

and $y = 0$ otherwise. The other activation function in use, log-softmax is a logarithm of a softmax. A softmax transforms a vector of values such that the sum of the elements is equal to 1. Using the following equation over the elements of a vector of length K.

$$Softmax(\vec{x}) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$$

To make a classification, the output size of the model is equal to the number of classes and the highest softmax value class is selected.

Finally, batch normalization and dropout are layers used to improve the training speed and reduce overfitting respectively. Batch normalization takes whatever input batch it is given and normalizes it, that is, the batch has its mean subtracted and is then divided by its standard deviation. This has the effect of making the model training faster in general. Dropout, on the other hand, reduces reliance on any given neuron directly upstream by, at random, multiplying the output of that neuron by 0 only in training. This process helps in avoiding overfitting to the training set and produces modes that fit better to the actual patterns in the data.

4.2 Training Methods

This section describes how the model architecture described in section 4.1 is trained to achieve the best performance possible. As was mentioned in section 3, one of the main challenges that training will encounter in this activity is the highly imbalanced dataset. This section will cover the basic training process as well as specific solutions implemented to address the data imbalance problem.

Initially, the dataset is split in train, validation and test datasets. They are made of 70%, 15% and 15% of all TDMs respectively, randomly chosen. Each dataset has its own purpose, the train dataset is used to train the models directly. The validation dataset is used to find the hyper parameters of the model that yield the best performance and, finally, the test set is used at the very end of the process to ensure that the model generalizes to a dataset to which it has not been ever exposed to.

The optimization process of the weights of the model has as objective the minimization of a loss function. For this application, the loss function used is negative log likelihood loss. To understand negative log likelihood loss, it is important to understand the output of the softmax function used in the model. Given N possible classes, the softmax outputs a vector with the probability that one sample corresponds to one class in each element. Then, for many samples, this probability can be defined as a matrix of size number of samples by number of classes, let this matrix be named P. Similarly, knowing the true label of a sample, a matrix Y can be defined with the same shape as $P. y_{i,k} = 1$ if sample i belongs to class k. Then, the loss function is defined in Equation 2, where N is the number of samples and K the number of classes.

$$L(Y, K) = -\frac{1}{N} \sum_{i}^{N-1} \sum_{k}^{K-1} y_{i,k} \log(p_{i,k})$$

The output produced by the loss is 0 for a perfect prediction and infinite for the opposite case. Onto this loss function, a weight associated with each class is added such that the effect of class imbalance is minimized. The weights for each class is proportional to the number of samples in the most numerous class divided by the number of samples of each class respectively. It is important to note that the loss function is only used by the model in training. In order to then understand the performance of a trained model, other metrics, such as the f1 score, accuracy and recall, are used.

Training takes place with an early stopping criterion of 20 epochs as monitored on the loss of a subset of the training set consisting of 20% of all TDMs in the training set with the remaining 80% used for actual training. Upon triggering of this mechanism, the model weights that performed better on the early stopping set are restored. This process should stop the model training at the point where its performance is maximum.

During training, weight decay is also used as a regularization measure, similar to dropout. Weight decay is a loss penalty due to the size of the weights in the system. It reduces reliance on any single neuron and leads to more robust models.

Finally, the hyper parameter space is explored in order to maximize the performance of the model. Models are scored by their f1 score on the validation set. More detailed analysis on the results is obtained by means of a confusion matrix.

In the very last case for each particular classification task, the model was trained and evaluated as if it was using more than one TDM from the same object to make a classification. To achieve this, an artificial dataset was created from each original dataset which contains several TDMs grouped together as if they were one single long TDM. The datasets are split evenly between one, two and three TDM combination, except for the validation dataset which is done separately to evaluate the effect of predictions with different numbers of TDMs.

5 RESULTS

5.1 4-Class Attitude Stability Classifier

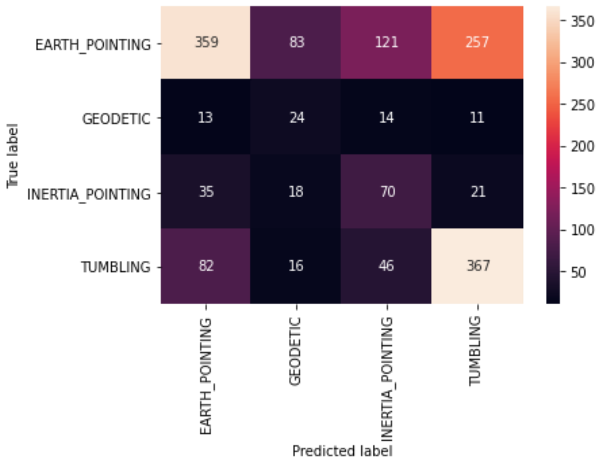
The first model in this category uses the following input feature vector: **[Normalized RCS, azimuth in sensor RF, elevation in sensor RF]** using single TDMs at a time. The best configuration of hyper parameters yielded an f1 score of 0.534 with a confusion matrix as shown in Figure 5-1.

Looking at the off diagonal elements in Figure 5-1, the model misclassifies Earth pointing objects as tumbling and vice versa. This is probably due to them being the two most populous classes. The next case of usual miss classification is concerning the difference between Earth pointing and inertia pointing. As expected it is difficult to distinguish these two classes.

Let's now consider the case for a classifier which uses, along with the normalized RCS, azimuth and elevation data from a geocentric inertial reference frame and a geocentric rotating reference frame. The feature vector then is: **[Normalized RCS, azimuth in sensor RF, elevation in sensor RF, azimuth in ITRF, elevation in ITRF, azimuth in GCRF and elevation in GCRF].** The best model in this case yielded an f1 score of 0.616 with the confusion matrix shown in Figure 5-2.

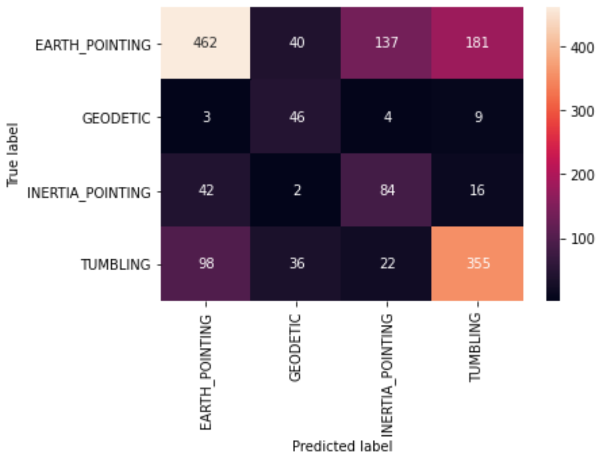
The origin of the performance gain is not clear in this case as misclassifications between Earth and Inertia pointing objects seem to increase. This case was expected to increase the most in performance. It seems as if the increase in information provided for the model allowed for better results overall.

In the last tested case, where the model is tested and trained using combinations of 1, 2 or 3 TDMs of the same object together with the same feature set as the last case, we obtain the results shown in Figure 5-3 on the validation set.



True label \ Predicted label	EARTH_POINTING	GEODETIC	INERTIA_POINTING	TUMBLING
EARTH_POINTING	359	83	121	257
GEODETIC	13	24	14	11
INERTIA_POINTING	35	18	70	21
TUMBLING	82	16	46	367

Figure 5-1. Confusion matrix for 4-class attitude stability classification with azimuth and elevation from sensor.



True label \ Predicted label	EARTH_POINTING	GEODETIC	INERTIA_POINTING	TUMBLING
EARTH_POINTING	462	40	137	181
GEODETIC	3	46	4	9
INERTIA_POINTING	42	2	84	16
TUMBLING	98	36	22	355

Figure 5-2. Confusion matrix for 4 class attitude stability classification with azimuth and elevation from sensor, GCRF and ITRF reference frames.

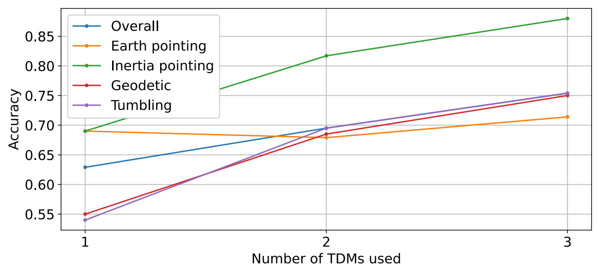


Figure 5-3. Accuracy of each classification class as a function of number of TDMs used.

As can be seen in Figure 5-3 there are significant performance improvements to be gained by using several TDMs from the same object to make a classification. In general, per class accuracy tends to increase as well as overall accuracy with the f1 score for one, two and three TDMs used is 0.629, 0.696 and 0.754. It will be the objective of successive studies to obtain the limits of this technique. Notice as well that the performance of this model using only one TDM is superior to that of the model trained using single TDMs. This is a consequence of the additional entropy introduced in the dataset. The TDMs themselves do not change but by arranging them in different ways much stronger regularization can be achieved.

Finally, it is interesting to show the evolution of the overall performance of the models through the iterations. This is shown in Table 5-1.

	F1 score
Sensor RF only	0.534
Sensor + GCRF ITRF	0.616
Sensor + GCRF ITRF 1 TDM	0.629
Sensor + GCRF ITRF 2 TDM	0.696
Sensor + GCRF ITRF 3 TDM	0.754

Table 5-1Results summary for 4-class attitude stability classification

5.2 3-Class Attitude Stability Classifier

In this section, the performance of the classifiers trained to distinguish between stable, geodetic and tumbling objects is presented. As was done in the previous section, the first model uses the following feature vector **[Normalized RCS, azimuth in sensor RF, elevation in sensor RF]** from one TDM to make the prediction, then, additional features will be included and finally several TDMs. The confusion matrix for the best classifier using only normalized RCS, azimuth and elevation from the sensor is shown in Figure 5-4.

Overall, the f1 score in this case was 0.635 on the validation set. From the confusion matrix in Figure 5-4, the model seems to have more misclassifications between tumbling and stable objects. However, results improve when using **[Normalized RCS, azimuth in sensor RF, elevation in sensor RF, azimuth in ITRF, elevation in ITRF, azimuth in GCRF and elevation in GCRF]** as feature vector, as shown in Figure 5-5. With these new feature, stable objects are classified as geodetic less often. The total f1 score in this case is 0.738.

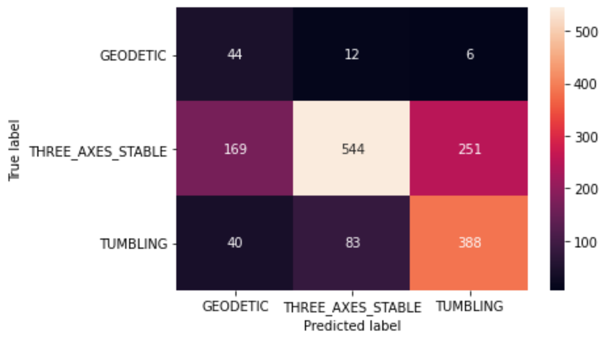


Figure 5-4. Confusion matrix for 3 class attitude stability classification with azimuth and elevation from sensor.

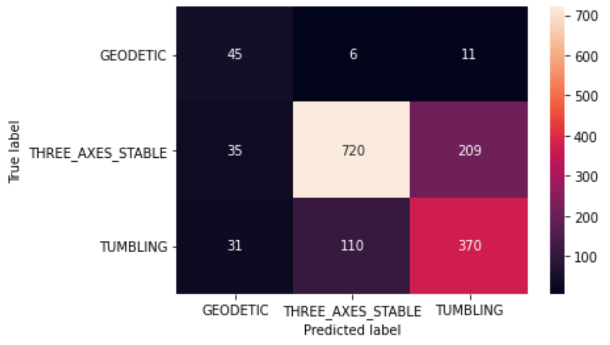


Figure 5-5. Confusion matrix for 3 class attitude stability classification with azimuth and elevation from sensor, GCRF and ITRF reference frames.

Finally, considering the case in which up to three TDMs are used to train the model, the results shown in Figure 5-6 are obtained. As is apparent in the figure, while the model was trained in up to three TDMs, evaluation has taken place with up to four TDMs to explore the behaviour outside the training range. The difference between training with single TDMs and multiple TDMs with three class attitude stability classification is very similar to the earlier shown four class attitude classification. Evaluating using only one TDM has an f1 score of 0.726, compared to 0.738. This small decrease in performance with one TDM is quickly surpassed by using more than one TDM for the prediction. The performance seems to be limited to as many TDMs as were used for training, as using four TDMs keeps the performance the same as with three. Moreover, in some cases such as for geodetic objects, performance drops significantly with 4 TDMs. Therefore, it can be concluded that performance is erratic when using more TDMs than the mode was trained on.

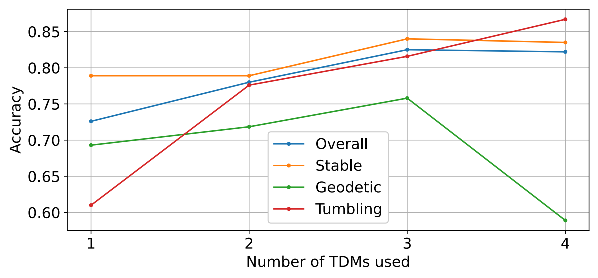


Figure 5-6. Accuracy of each classification class as a function of number of TDMs used.

As was done in the previous section, a visualization of the maximum performance achieved in this task is shown in Table 5-2.

	F1 score
Sensor RF only	0.635
Sensor + GCRF ITRF	0.738
Sensor + GCRF ITRF 1 TDM	0.726
Sensor + GCRF ITRF 2 TDM	0.78
Sensor + GCRF ITRF 3 TDM	0.825
Sensor + GCRF ITRF 4 TDM	0.822

Table 5-2 Results summary for 3-class attitude stability classification.

5.3 Direct Object Identification

In this section the results obtained from the classification of individual objects in LEO are shown and analysed. The problem is radically different from the other two as the number of classes is one order of magnitude larger. However, the same techniques generalize to such large number of classes. The analysis will be structured in a similar fashion to the other two, with features being added sequentially from simple to complex.

Using the model described in section 4, with **[Normalized RCS, azimuth in sensor RF, elevation in sensor RF]** as feature vector, the best f1 score found is 0.325, with the confusion matrix shown in Figure 5-7.

Some interesting features can be observed in Figure 5-7, for instance, notice how the model is mainly recognizing platforms instead of actual objects. That is, swarm A B and C are mixed together regularly. The same happens for some Sentinel, Metop and Jason groups for instance.

This is the case as well for the cosmos satellites. All of them are calibration spheres which are very similar to each other resulting in the spread observed in the figure. Introducing the feature vector: **[Normalized RCS, azimuth in sensor RF, elevation in sensor RF, azimuth in ITRF, elevation in ITRF, azimuth in GCRF and elevation in GCRF]**, the f1 score increases to 0.379, with the resulting confusion matrix shown in Figure 5-8.

There is not much difference between Figure 5-7 and Figure 5-8 visually. However, there is a significant performance increase judging by the f1 score comparison. Finally, training on inputs of up to three TDMs yielded the performance shown in Figure 5-9.

In the figure, it is clear that there are massive performance increases to be had by using several TDMs, a nearly 0.2 increase in the f1 score from using one to four TDMs.

As was done in the previous section, a visualization of the maximum performance achieved in this task is shown in Table 5-3. It is particularly interesting to point out that the gap in performance achieved by using more than one TDM is much larger than in the other two cases. Suggesting that the additional TDMs are helping complete the histogram of RCS values that characterizes a particular object.

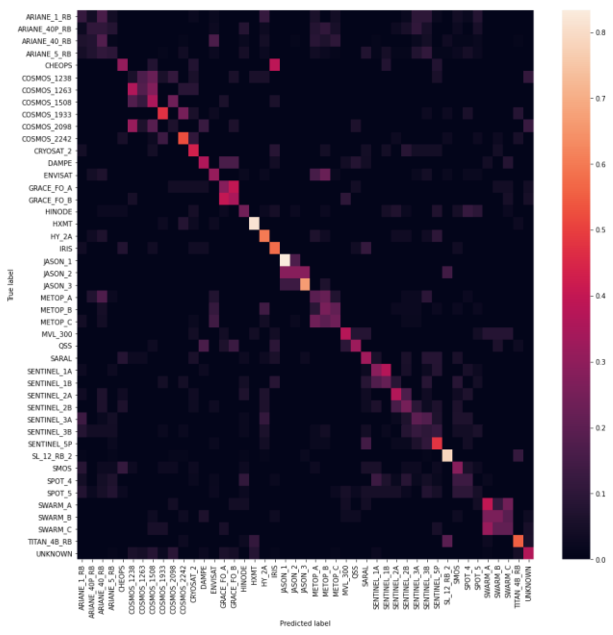


Figure 5-7. Confusion matrix for object identification with azimuth and elevation from sensor.

Looking at the confusion matrices in Figure 5-7 and Figure 5-8, it is clear that it is difficult for the model to distinguish some of the RSOs because they share a common platform. In the future, analysis the object identification task may evolve into a platform identification task, which will probably prove to be more successful.

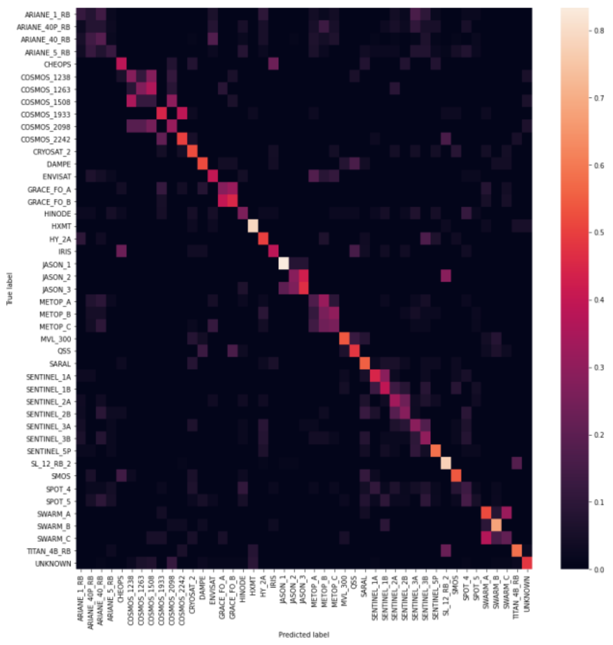


Figure 5-8. Confusion matrix for object identification with azimuth and elevation from sensor, GCRF and ITRF reference frames.

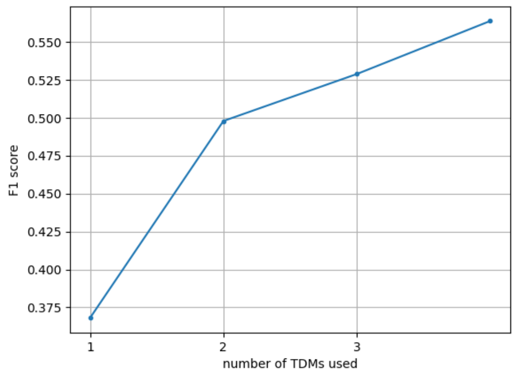


Figure 5-9. F1 score as a function of number of TDMs used.

	F1 score
Sensor RF only	0.325
Sensor + GCRF ITRF	0.379
Sensor + GCRF ITRF 1 TDM	0.368
Sensor + GCRF ITRF 2 TDM	0.498
Sensor + GCRF ITRF 3 TDM	0.529
Sensor + GCRF ITRF 4 TDM	0.564

Table 5-3 Results summary for direct object identification

6 CONCLUSION AND CONTINUED WORK

In this work, a tool to identify characteristics of RSOs based on real RCS data obtained from S3TSR radar has been proposed. The tool exploits established ML techniques like the LSTM to make the classification. Even though the dataset available is limited in size, remarkable results have been obtained showing that these techniques have a promising potential if refined further.

For attitude classification tasks, accuracies of 62% for four class classification and 72% for three class classification are achieved with a single TDM. However, accuracy increases greatly when introducing more TDMs reaching values of 75% and 82% respectively.

In direct object classification, results are promising but reflect the increased difficulty of the problem. Accuracies here reach 37% with one TDM and up to 56% using four.

These results show the viability of these methods as a way to first identify whether an object is or remains stable or as a backup to the traditional track correlation method to identify objects.

However, these results are still preliminary as new features are being developed to help the classifiers and additional data is being used to improve the models. Deep Learning models, such as the ones presented here, have shown their ability to increase their performance continuously when additional data is made available for training. All in all, results are expected to get better with future iterations of the tool.

7 REFERENCES

1. X. J. Fu, M. Wang, C. Wang, and M. G. Gao. (2010). Attitude Stability Determination Based on RCS Series Information Extraction for Space Objects, *AMM*, vol. 20–23, pp. 936–941.

2. Y. Li, W. Qu, Z. Wang, and Y. Zhao (2016). Space Target Motion Attitude Discrimination based on RCS Time Sequence, *ICSPS 2016: Proceedings of the 8th International Conference on Signal Processing Systems*.

3. J. Qin, J. Zhu, H. Peng, T. Sun, and D. Hu. (2018). The Estimation of Satellite Attitude Using the Radar Cross Section Sequence and Particle Swarm Optimization. *IEICE Trans. Fundamentals*, vol. E101.A, no. 3, pp. 595–599.

4. W.-J. Zhong, J.-S. Wang, W.-J. Ji, X. Lei, and X.-W. Zhou. (2015). The attitude estimation of three-axis stabilized satellites using hybrid particle swarm optimization combined with radar cross section precise prediction. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 230, no. 4, pp. 713–725.

5. T. Wang, W. Bi, Y. Zhao, and W. Xue. (2015). Radar target recognition algorithm based on RCS observation sequence — set-valued identification method. *J Syst Sci Complex*, vol. 29, no. 3, pp. 573–588.

6. D. Mehrholz. (2001). Radar techniques for the characterization of meter-sized objects in space. *Advances in Space Research*, vol. 28, no. 9, pp. 1259–1268.

7. Xuehui Lei, Xiongjun Fu, Cai Wang and Meiguo Gao. (2011). Statistical feature selection of narrowband RCS sequence based on greedy algorithm, *2011 IEEE CIE International Conference on Radar (Radar)*.

8. X. Liu, M. Gao, and X. Fu. (2006). A Nearest Neighbor Fuzzy Classifier for Radar Target Recognition Using Combined Features. *2006 8th international Conference on Signal Processing*.

9. C. Wang, X. Fu, Q. Zhang, L. Jiao, and M. Gao. (2012). Space object identification based on narrowband radar cross section. *2012 5th International Congress on Image and Signal Processing (CISP)*.

10. J. Tkac, S. Spirko, and L. Boka. (2000). Radar object recognition by wavelet transform and neural network. *13th International Conference on Microwaves, Radar and Wireless Communications. MIKON-2000*.

11. David K. Barton, David Brillinger, A. H. El-Shaarawi, Patrick McDaniel, Kenneth H. Pollock, Michael T. Tuley. (1998). Final report of the haystack orbital debris data review panel. *Technical Memorandum 4809，NASA*, pp.11-14.

12. Luzum, B., & Petit, G. (2012). The IERS Conventions (2010): Reference systems and new models. *Proceedings of the International Astronomical Union*, *10*(H16), 227-228.

13. Kaplan, G. H. (2006). The IAU resolutions on astronomical reference systems, time scales, and Earth rotation models. *arXiv preprint astro-ph/0602086*.