Executive Summary of the Thesis

Space object identification and correlation through AI-aided light curve feature extraction

Laurea Magistrale in Space Engineering - Ingegneria Spaziale

Author: Chiara Bertolini

Advisor: Prof. Pierluigi Di Lizia

Co-advisors: Prof. Mauro Massari, PhD Candidates Andrea De Vittori, Riccardo Cipollone and Luca Facchini

Academic year: 2021-2022

1. Introduction

In recent years, as the space economy opened the doors for more and more commercial applications, the near-Earth environment has gotten even more populated [1]. Therefore, this phenomenon gives awareness that the space environment has a limited capacity: in fact, as more objects are sent in-orbit, the risk of collisions and successive fragmentations increases. As a consequence, over the last decades, it has arisen the necessity to monitor this ever-growing population and assess the in-orbit collision and fragmentation risk. The survey is performed by space agencies through the Space Surveillance and Tracking (SST) systems [2].
Ground-based stations allow to retrieve orbital data of human-made objects. In particular, thanks to optical telescopes and, subsequently, photometry analysis, light curves can be retrieved. Light curves are defined as the variation in time of the objects' brightness; they encapsulate information such as the orbit regime, the tumbling motion and the geometry of the observed spacecraft. By analysing light curves, it is therefore possible to characterise the observed space object.

In general, analytical methods, like the so-called Light Curve Inversion, have been extensively used for the identification of space objects [3]. However, complex models have to be considered and the resulting analysis is computationally time-consuming.
Consequently, the state of the art is now drifting to the use of machine learning with bespoke Convolutional Neural Networks (CNN) or Recursive Neural Networks (RNN) ensuring up to 90% prediction accuracy [4] [5]. Light curves provided in these works are obtained from objects in the Geostationary Orbit (GEO) belt, with ad hoc reflection models and simple platform geometries [4].
This thesis project proposes a novel approach to light curve characterization through the Machine Learning based Light curve Analysis (ARIEL) tool.
Raw light curves are recovered from the database managed by the Mini-MegaTORTORA (MMT-9) observatory [6]. Among all space objects within the catalogue, ARIEL only applies to those in the Low Earth

Orbit (LEO) region, featuring periodic or aperiodic tumbling motion. They are also classified into Rocket body, Debris and Satellite type.
Raw data is then pre-processed and then fed into three different NN:

ROGUE focuses on the identification of the Rocket body category among a pool of other resident space objects.

LINDEN compares two input light curves by means of a Feature extraction and Correlation block.

SIERRA performs a similarity analysis through a Siamese Network [7].

Performances for these networks are then assessed using different datasets obtained by varying the number of spacecraft platforms.

2. Theoretical background

Light curves

As above-mentioned, light curves are recovered from shots acquired with optical telescopes. An example of the observatory is represented by the Mini-MegaTORTORA system. This observatory has been operational since 2014, and it is the result of a collaboration between Kazan Federal University (RUS), Special Astrophysical Observatory of the Russian Academy of Sciences and Parallax Ltd [6]. The MMT-9 catalogue predisposes two constantly updated databases: one for natural transients, i.e. asteroids, while the other for space debris and other for human-made space objects.
The latter is employed extensively in this project. The main informations recovered are:

- Name of the spacecraft and corresponding ID
- Type of object: Rocket body, Debris and Satellite
- Variability: attitude regime of the object which can be Periodic, Aperiodic or Non-variable
  If Periodic, the Variability period is specified
- Date and Time of acquisition for each Track considered
- Apparent magnitude at each time instant for each Track considered
- Track ID each light curve is assigned to

Focusing on light curve characterization, their apparent magnitude can be obtained from the apparent luminosity of the source through a logaritmic as Equation 1 shows:

$$m_2 - m_1 = -2.5 \log \left(\frac{l_1}{l_2} \right) \tag{1}$$

where 1 and 2 represent two distinct objects with luminosity l_1 and l_2 respectively.
Figure 1 depicts an example of light curve recovered from the MMT-9 database.

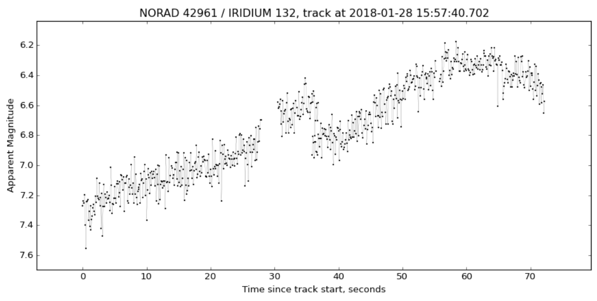


Figure 1: Example of light curve [6]

Before entering the NN, however, these raw light curves are pre-processed using a moving average, the Savitzky-Golay filter [8]. The smoothing properties of this filter allow to reduce high frequency noise. The outcome can be seen in Figure 2, where the grey signal is the raw light curve, while the red is the filtered one.

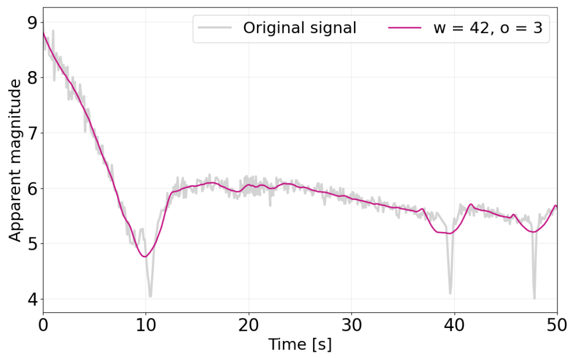


Figure 2: Filtering results (cropped light curve)

Deep learning fundamentals

Deep learning networks are a subset of Machine learning models, which consider the so-called hidden layers between the input and output ones. Each hidden layer is defined by a fixed number of units, called neurons, activated depending on the incoming connections. These connections are defined in terms of weights and

biases.
The combination of input, output and hidden layers form a Fully-Connected network. Different NN structures can also be employed such as CNN and RNN.
CNNs stack up as Convolutional layers. They perform a convolution operation between the input tensor and a kernel one, sliding on top of the former with a given stride. This operation allows to retain the main features of the input.
Regarding RNN, the most common structure is the Long-Short Term Memory (LSTM) cells. In this case, sequential data is analysed preserving or discarding the information gathered from previous cells with dedicated gates.
After the NN setup, it needs to be trained. The aim is to find weights and biases such that the predictions made match the labels via a minimization process involving a loss function. The shortlisted cost function is the Cross Entropy loss, as it is suited for class probability prediction. As the minimization proceeds, weights ans biases are updated using gradient-based algorithms, such as the Adam method, and a hyperparameter, the learning rate. This parameter needs to be tuned accordingly to improve the training process. Last but not least, an additional metrics can be considered during training to have another performance indicator: the most common one is Accuracy.
Particular attention has to be given in the model structure and dataset provided to avoid over- or underfitting of the network.

Siamese networks have a slightly different architecture than the prior-mentioned models, called Sequential, since each layer is stacked on top of the other. In this case, the overall dataset is divided in Anchor, the reference, Positive and Negative, the closest and the farthest prediction from the reference.
Then, an embedding model extracts features from the inputs. The network evaluates the distance Anchor-Positive and Anchor-Negative in order to bring the former closer and the latter farther. Thus, a specific loss function is employed: the Triplet Loss.
Instead of Accuracy, Cosine Similarity evaluates the predictions made by the embedding model.

3. Architectures

Three different architectures are developed inside the ARIEL framework: ROGUE, LINDEN and SIERRA.

ROGUE

The **R**ocket b**o**dies Li**g**ht c**u**rves Id**e**ntification (ROGUE) network has aims at recognizing Rocket bodies among light curves of different types. The structure is a combination of CNN and LSTM cells, as can be seen from Figure 3.

Layer	Features	Number of parameters
Embedding	Input dimension = 1.000; Output line = 20; Activated masking	20.000
1D Convolutional	Filters = 64; Kernel size = 3; Padding = same	3.904
Batch Normalisation		256
ReLU activation		
1D Convolutional	Filters = 64; Kernel size = 3; Padding = same	12.352
Batch Normalisation		256
ReLU activation		
LSTM cell	Units = 64; Return sequences = True	33.024
Batch Normalisation		256
LSTM cell	Units = 32; Return sequences = True	12.416
Global Max Pooling (1D)		
Fully-Connected	Units = 10; Activation = ReLU; L2 regularization applied with $\lambda = 0.001$	330
Fully-Connected	Unit = 1	11

Figure 3: ROGUE NN breakdown structure

This test is conceived to verify the capability of the NN to identify a defined category of spacecraft.

LINDEN

The **L**ight curve **I**de**n**tification an**d** Corr**e**latio**n** (LINDEN) NN compares two light curves and determines the correlation degree among the twos. The main contribution of this strategy is the Correlation block shown in Figure 4.
Two models have to be therefore developed:

- the Feature extraction part analyses the light curves and predicts the objects' type
- the Correlation evaluation block which, given the above-mentioned predictions, evaluates the distance between them.

The Feature extraction model is an improved

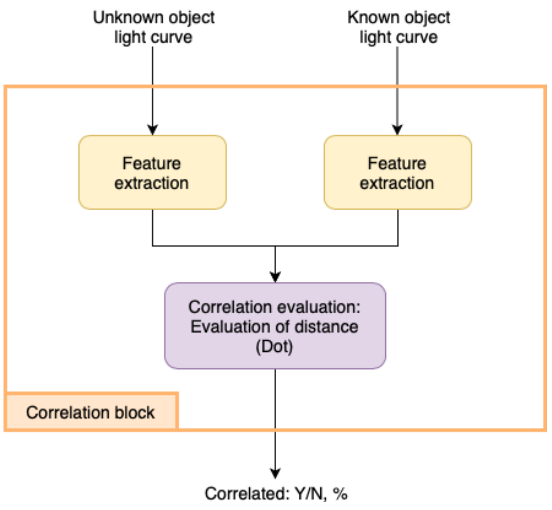


Figure 4: LINDEN Structure

version of the ROGUE model, as shown in Figure 5.

Layer	Features	Number of parameters
Embedding	Input dimension = 1.000; Output line = 20; Activated masking	20.000
1D Convolutional	Filters = 64; Kernel size = 3; Padding = same	3.904
Batch Normalisation		256
ReLU activation		
1D Convolutional	Filters = 64; Kernel size = 3; Padding = same	12.352
Batch Normalisation		256
ReLU activation		
1D Convolutional	Filters = 64; Kernel size = 3; Padding = same	12.352
Batch Normalisation		256
ReLU activation		
Max Pooling (1D)		
LSTM cell	Units = 64; Return sequences = True	33.024
Batch Normalisation		256
LSTM cell	Units = 64; Return sequences = True	33.024
Batch Normalisation		256
LSTM cell	Units = 32; Return sequences = True	12.416
Global Max Pooling (1D)		
Fully-Connected	Units = 10; Activation = ReLU; L2 regularization applied with $\lambda = 0.001$	330
Fully-Connected	Units = 7; Activation = Softmax	77
Fully-Connected	Units = 3; Activation = Softmax	24

Figure 5: LINDEN - Feature Extraction NN breakdown structure

The output gives a prediction vector over the class labels.
After having trained the Feature extraction part, it is inserted in the overall LINDEN Correlation block. Finally, the correlation between the prediction vectors is performed.
This final part is done thanks to a Dot layer with activated normalization, providing the angular distance between the predictions with a simple scalar multiplication. As the Feature extraction model is frozen within the Correlation block, this allows to compute a correlation degree without being influenced by uncertainties in the model.

SIERRA

Siamese Network **L**i**g**ht curv**es** Co**rr**el**a**tion (SIERRA) is a Siamese Network as shown in Figure 6.

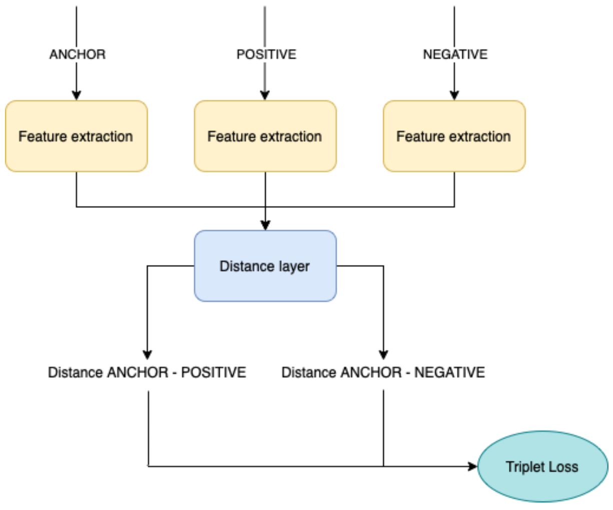


Figure 6: SIERRA NN breakdown structure

It encompasses the above-mentioned Feature extraction block as embedding model. With a successive custom Distance layer, it outputs the Anchor-Positive and Anchor-Negative distances. Those are then fed into the Triplet Loss during the training process, so as to minimise the former while maximising the latter.

Network setup

To assess the performances of the ARIEL networks, light curves with the following criterion have been considered:

- The objects considered belong to LEO or Low-Medium Orbit (LMO) regions
- Their tumbling motion is periodic or aperi-

odic
- They have a sufficient number of light curves

Light curves that respect these conditions have been filtered and stored in datasets, with additional informations like the name and the type of object.
Moreover some layers require the inputs to have a fixed length. Therefore, light curves are zero-padded so that they all have the same dimensions. Those additional values are not considered in the NNs thanks to the Activated masking in the Embedding layer, the first layer of both ROGUE and Feature extraction block from LINDEN and SIERRA.
A problem that might also rise is that some datasets comprise a scarce number of light curves, hence limiting the training performances. Thus, the dataset is "multiplied", e.g. it is appended up to four times into the same file.
Different datasets have been considered and obtained mainly by varying the number of platforms available.
First, consider nominal conditions of operation of ARIEL:

Nominal_1 (Nom_1) includes only periodic objects and one platform per type, i.e. only three objects per type. 434 light curves are contained, therefore the dataset is "multiplied" 4 times.

Nominal_2 (Nom_2) features both periodic and aperiodic objects and one platform per type and variability; therefore six objects are considered with a total number of 645 light curves, which are thus "multiplied" 4 times

Since these datasets have been used for the three ARIEL architectures, a slight unbalance towards the Rocket Body type can be perceived in the results.
Then, a variability test assesses the extent of ARIEL capabilities. All the objects considered have periodic or aperiodic attitude regime.

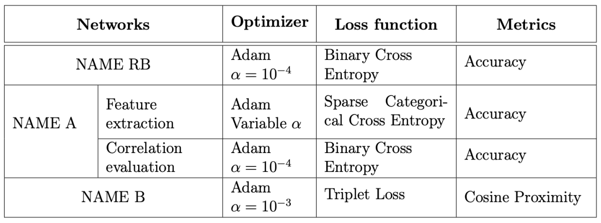
Variability_1 (Var_1) has 72 objects: 19 Rocket bodies, 37 Satellites and 16 Debris. The overall number of light curves is 4334.

Variability_2 (Var_2) considers 40 objects: 9 Rocket bodies, 19 Satellites and 12 Debris. It includes 1800 light curves and is s "multiplied" 2 times.

Variability_3 (Var_3) contains 20 objects: 6 Rocket bodies, 8 Satellites and 6 Debris, comprising 720 light curves and is "multiplied" 4 times

The number of platforms is not uniform among the three types considered, nevertheless, the number of light curves associated to each type is balanced out. Eventually the datasets are shuffled and divided in Training, Testing and Validation subsets considering 60%-20%-20% ratios.

The last step involves the choice of optimizers, loss functions and metrics, appropriate for each ARIEL architecture. These technical aspects are summarized in Figure 7.



Networks		Optimizer	Loss function	Metrics
NAME RB		Adam $\alpha = 10^{-4}$	Binary Cross Entropy	Accuracy
NAME A	Feature extraction	Adam Variable α	Sparse Categorical Cross Entropy	Accuracy
	Correlation evaluation	Adam $\alpha = 10^{-4}$	Binary Cross Entropy	Accuracy
NAME B		Adam $\alpha = 10^{-3}$	Triplet Loss	Cosine Proximity

Figure 7: Optimization problem setup

A variable learning rate α is considered in the Feature extraction block: a staircase-like variation is imposed, ranging from 10^{-3} to $5 \cdot 10^{-5}$.

4. Results

Hereafter are summarized the results for the ARIEL networks, obtained considering the above-mentioned datasets.
The training has been performed using Google Colaboratory. Due to the limited GPU time availability, the training has been divided in sessions from 100 to 200 epochs.
A special test case is also considered for LINDEN Feature extraction: the Debris test case.

ROGUE - Results

In Table 1 all the final performance metrics obtained for each dataset are summarized and information on the training, if separated segments have been considered.
In general Variability datasets have lower performances with respect to Nominal ones. Only Var_3 has similar results than Nom_2, showing that ROGUE can best differentiate the Rocket body type among up to 20 different platforms.
However Nominal datasets do not present over-

Dataset	Loss	Accuracy	Training
Nom_1	~ 0.1	$\sim 98\%$	
Nom_2	~ 0.1	$\sim 97\%$	2 training portions of 200 and 100 epochs
Var_1	≥ 0.5 ; ≤ 0.56	$\geq 71\%$; $\leq 74\%$	3 training portions of 200 epochs
Var_2	≥ 0.23 ; ≤ 0.3	$\geq 89\%$; $\leq 92\%$	8 training portions of 200 epochs
Var_3	≥ 0.15 ; ≤ 0.2	$\geq 93\%$; $\leq 95\%$	3 training portions of 100 and 200 epochs

Table 1: Performances for ROGUE

fitting as Variability sets do. In particular, a 6% difference can be perceived between Training and Validation loss for Var_1, reducing to 3% and 2% for the subsequent datasets. The difference in Accuracy is not as significant, remaining around 3%.

LINDEN - Results

As previously mentioned, the Feature extraction block is trained separately and then inserted in the overall Correlation block.

Feature extraction

The performances obtained can be seen in Table 2.
The Var_2 and Var_3 datasets have comparable performance with respect to Nom_2, however they are still affected by a relevant overfitting. The main difficulty this phase faced was while considering the Var_1 dataset: in fact several modifications have been performed, none of which produced satisfactory results. Moreover, the confusion matrix associated to this training session, displayed in Figure 8, shows a peculiar aspect: the Debris type is completely missing among output predictions. This enforces the idea that Debris are more difficult to categorize due to their formation.
The Var_1 dataset therefore has not been considered in the training of the second part of LINDEN nor of SIERRA.

Dataset	Loss	Accuracy	Training
Nom_1	~ 0.1	$\sim 98\%$	
Nom_2	~ 0.1	$\sim 95\%$	
Var_1	~ 1.0	$\geq 50\%$; $\leq 53\%$	Not possible
Var_2	≥ 0.24 ; ≤ 0.3	$\geq 86\%$; $\leq 90\%$	3 training portions of 200 epochs
Var_3	≥ 0.15 ; ≤ 0.25	$\geq 90\%$; $\leq 94\%$	4 training portions of 200 and 100 epochs

Table 2: Performances for LINDEN - Feature extraction

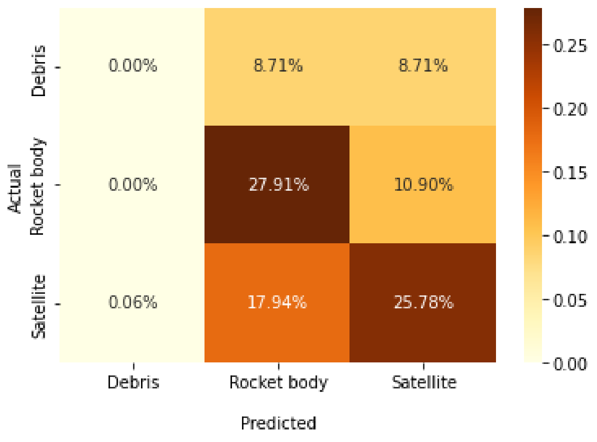


Figure 8: Confusion matrix for LINDEN - Feature extraction - Var_1

Correlation block

Var_1 was not considered for the overall block training, due to its poor performances, as shown above.
The performances for the remaining datasets are displayed in Table 3.
As the levels reached in the Feature extraction block leave room for uncertainty, the results obtained in the Correlation block are quite scarce, except for Nom_1.
Overfitting is observed also for Nom_2 becoming even more relevant for the Variability datasets, in particular for Var_2. However, the confusion observed is still below the 10% bound.
All in all, LINDEN proves its capabilities by granting an accurate type recognition, which most of the time allows correlation between the

Dataset	Loss	Accuracy	Training
Nom_1	≥ 0.04 ; ≤ 0.08	$\sim 99\%$	2 portions of 100 epochs
Nom_2	≥ 0.16 ; ≤ 0.32	$\geq 90\%$; $\leq 94\%$	3 portions of 50 epochs
Var_2	≥ 0.4 ; ≤ 0.7	$\geq 75\%$; $\leq 87\%$	
Var_3	≥ 0.25 ; ≤ 0.35	$\geq 87\%$; $\leq 90\%$	2 training portions of 100 epochs

Table 3: Performances for LINDEN - Correlation evaluation

two inputs to be established properly. However only up to 20 platforms can be considered at the same time in order to obtain accurate results.

SIERRA - Results

Even for SIERRA, Var_1 is not considered for training. Therefore only performances related to Nominal datasets, Var_2 and Var_3 are assessed and summarized in Table 4.
A single training session of 100 or 50 epochs is considered per dataset. It is worth mentioning that the metric employed is the Cosine Similarity.

Dataset	Loss	Cosine Similarity
Nom_1	< 0.01	Positive: ~ 0.99 Negative: ~ 0.5
Nom_2	≥ 0.04 ; ≤ 0.06	Positive: ~ 0.89 Negative: ~ 0.52
Var_2	≥ 0.07 ; ≤ 0.16	Positive: ~ 0.89 Negative: ~ 0.55
Var_3	≥ 0.08 ; ≤ 0.11	Positive: ~ 0.87 Negative: ~ 0.78

Table 4: Performances for SIERRA

As expected, the overfitting present in the Feature extraction block propagates to SIERRA, giving a 10% difference between Training and Validation loss for Var_2.
The similarity obtained in the different datasets is roughly giving the same gap, therefore the Positive and Negative outcomes are properly distinguished.

The only case showing remarkable confusion is with Var_3. It may be due to the fact that Anchor and Negative have common characteristics not considered during the Feature extraction block.

Debris test case

An additional test is performed, focusing on the recognition on the Debris type. This type may be one of the reasons for the confusion observed in many test cases. In fact, these objects can be unused spacecrafts or rocket body parts. In those cases, the light curves resulting are similar to the spacecrafts classified as Rocket body or Satellite. Thus the confusion might be resolved by eliminating this type during the training phase, and afterwards verifying the origin of the Debris-type objects considered.
Therefore two dedicated datasets have been considered and tested:

No_Debris_training (NoDeb_train) comprises only of Rocket bodies and Satellites, therefore 21 objects are considered: 7 Rocket bodies and 14 Satellites, for a total of 1200 light curves, "multiplied" 2 times. This dataset is employed only for Training and Validation.

Debris_test (Deb_test) comprises only Debris, labeled as their origin - Rocket body or Satellite. 16 platforms are considered: 7 Rocket bodies and 9 Satellites. This dataset, containing 820 light curves, is used for Testing only.

This analysis is performed only on LINDEN Feature extraction block. After performing two training sessions of 200 epochs on NoDeb_train, the metrics obtained are:
- **Loss**: ≥ 0.15 ; ≤ 0.3
- **Accuracy**: $\geq 90\%$; $\leq 95\%$

A significant overfitting is still present and the performances are in the order of Var_3, even if the convergence is obtained faster.
The model is then tested using the Deb_test dataset. The results are shown in the confusion matrix in Figure 9.
It indicates that the Feature extraction model still requires some training to correctly pin-point the origin of Debris. However, it might be the case that Deb_test are also includes fragmentation debris which are difficult to relate to the original object, thus hindering the type identifi-

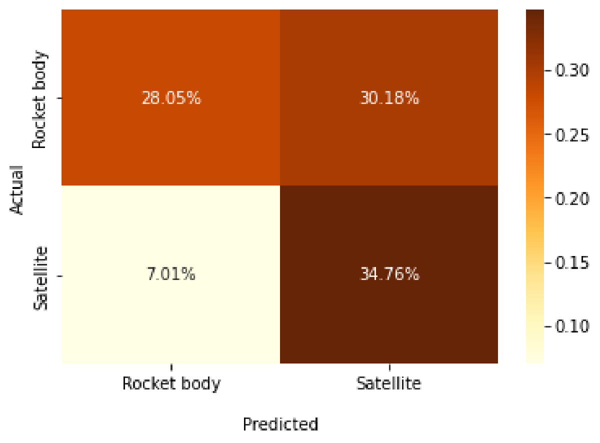


Figure 9: Confusion matrix for the Debris test

cation.

5. Conclusions

As the issue of the ever-growing space population is gaining more and more importance in the Space sector, it is of paramount importance to recognise the objects observed during surveys.
ARIEL provides a strategy to identify objects according to their type and to establish a degree of correlation between the unknown object and a catalogued one. This is done by the analysis of light curves through machine-learning based systems. The NN considered combines the capabilities of CNN and of LSTM: the former allowing to extract general characteristics while the latter focuses on time-dependent features.
Three architectures are thus proposed, each focusing on a different aspect of the problem at-hand:

ROGUE recognises the Rocket body type among a pool of space objects

LINDEN classifies two light curves according to the spacecraft type and infers whether they belong to the same category

SIERRA adapts a Siamese network to evaluate the distances between three objects' light curves

The light curves are obtained from the MMT database and have been pre-processed, in particular filtered with the Savitzky-Golay smoothing filter.
After extensive training using different datasets, the performances have been assessed. In particular it can be observed that the loss reaches convergence and the resulting accuracy is around 90%. The significant gap observed for the similarity in SIERRA is proof that these networks predict the type of object with little confusion, which is in fact bounded below 10% in most of the cases.
However these NN are limited by datasets including diverse platforms: accurate type recognition is hampered, thus preventing the correlation to be performed.
Moreover, overfitting is omnipresent: in some cases it becomes substantial, therefore impacting the accuracy of the predictions done.
Some options can hence be proposed to improve ARIEL, e.g. consider a smaller number of different platforms or restrict the problem to the recognition of platforms among a same type, or a same attitude regime (i.e. periodic, aperiodic or non variable).
It may be also of interest to focus on the problem of the Debris type recognition: it is in fact the most mistaken, as some of these objects are unused satellites or intact rocket body parts. Therefore, a dedicated analysis among Debris may be needed if those objects are involved.

6. Acknowledgements

I would first like to extend my gratitude to my advisor, Prof. P Di Lizia, for offering me this thesis project and always followed it with great interest. I would also like to pay my special regards to Prof. M. Massari, Andrea de Vittori, Riccardo Cipollone and Luca Facchini for guiding me throughout the thesis, providing me with precious advices and insights, and for their constant readiness to help. I would also like to acknowledge the MMT-9 observatory for their incredibly useful database, fundamental for this project.

References

[1] ESA Space Debris Office, “ESA annual space environment report,” Tech. Rep. 6, European Space Agency, Darmstadt, Germany, April 2022.

[2] EU SST, “European space surveillance and tracking,” 2016. `https://www.eusst.eu`.

[3] B. Bradley and P. Axelrad, “Lightcurve inversion for shape estimation of GEO objects from space-based sensors,” *ISSFD*, 2014.

[4] R. Linares and R. Furfaro, "Space object classification using deep convolutional neural networks," *IEEE*, 2016.

[5] E. Kerr, G. Falco, N. Maric, D. Petit, P. Talon, E. Geistere Petersen, C. Dorn, S. Eves, N. Sánchez-Ortiz, R. Dominguez Gonzalez, and J. Nomen-Torres, "Light curves for GEO object characterisation," (Darmstadt, Germany), ESA, ESA Space Debris Office, 4 2021. Proc. 8th European Conference on Space Debris.

[6] S. Karpov, E. Katkova, G. Beskin, A. Biryukov, S. Bondar, E. Davydov, E. Ivanov, A. Perkov, and V. Sasyuk, "Massive photometry of low altitude artificial satellites on Mini-Mega Tortora," 2015. Database: `http://mmt.favor2.info/satellites`.

[7] H. Essam and S. Valdarrama, "Image similarity estimation using a Siamese Network with a Triplet loss," 2021. `https://keras.io/examples/vision/siamese_network/`.

[8] N. Gallagher, "Savitzky-golay smoothing and differentiation filter," 2020. `https://eigenvector.com/wp-content/uploads/2020/01/SavitzkyGolay.pdf`.

Space object identification and correlation through AI-aided light curve feature extraction

Tesi di Laurea Magistrale in
Space Engineering - Ingegneria Spaziale

Author: **Chiara Bertolini**

Student ID: 952823
Advisor: Prof. Pierluigi Dilizia
Co-advisors: Prof. Mauro Massari, Andrea De Vittori, Riccardo Cipollone,
Luca Facchini
Academic Year: 2021-22

Abstract

The space environment in the past years has been getting more and more crowded. In fact, commercial applications are getting of interest for businesses or everyday-life purposes. This phenomenon however gave the awareness of the space debris problem: the fact that more objects are sent in orbit is directly connected to the formation of new potential inactive debris through for example collisions with other spacecrafts. These debris therefore represent a threat for active satellites.
Hence ad-hoc sensor networks monitor this ever-growing population, which must be tracked and catalogued. This operation is done by Space Surveillance and Tracking (SST) systems which allow to assess the in-orbit collision risk as well as characterise fragmentations. Within this context, optical ground stations can spot objects in different orbital regimes and retrieve valuable information, such as the target light curve.
In particular, through optical telescopes, the objects light curves can be retrieved. Light curves are the variation in time of the spacecraft brightness. These data encapsulate the object tumbling motion, orbit regime and geometry information. The state-of-the-art approaches to light curve analysis may be computationally expensive and fail in linking one object to a platform. In fact, complex models are employed. Those particular strategies need an accurate model of the geometry analysed and the simplest formulation is optimized for convex objects. If other geometries are contemplated, the recognition fails entirely.
Thus, other strategies can come in handy, for example machine learning. The capabilities of Neural Network (NN) have been proved in many sectors - including the space one - and for all sort of applications, like image processing or speech recognition. These systems are capable of finding an input-output relationship without an explicit formulation. They "learn" features through training of the model.
Thanks to this vast range of applications, it may be possible not only to directly identify the object, but also to relate an unknown light curve to a known object.

In this dissertation, the M**a**chine Lea**r**ning based L**i**ght curv**e** Ana**l**ysis (ARIEL) tool is presented and declined in three sub NNs:

- Identification of a type of object using **R**ocket b**o**dies Li**g**ht c**u**rves Id**e**ntification

(ROGUE)

- Feature extraction and correlation evaluation through **L**ight curve **I**de**n**tification an**d** Corr**e**latio**n** (LINDEN)
- Similarity evaluation with **S**iamese Network L**i**ght curv**e**s Co**rr**el**a**tion (SIERRA)

These networks employ a combination of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) to extract general and time-dependent features from the input light curves.
Light curves are recovered from the public database handled by the Mini-MegaTORTORA (MMT-9) observatory. Afterwards, a pre-processing phase lowers the noise levels thanks to Savitzky-Golay (SavGol) filter.
The three models are then extensively trained and assessed in terms of training capabilities and predictions accuracy by varying the dataset.
The attained performance confirms that NNs can easily extract features and evaluate the correlation between two objects. However, this strategy is limited by the number of different platforms considered. In fact, if it becomes substantial, the three models have some difficulties in accomplishing their task.

Keywords: light curves, neural networks, correlation

Abstract in lingua italiana

L'ambiente spaziale negli ultimi anni è diventato sempre più affollato, in particolare a causa di applicazioni commerciali sempre più d'interesse per le imprese o la vita di tutti i giorni. Questo fenomeno tuttavia ha dato la consapevolezza del problema dei detriti spaziali: il fatto che più oggetti vengono inviati in orbita è direttamente collegato alla formazione di nuovi potenziali detriti inattivi attraverso ad esempio collisioni con altri veicoli spaziali. Questi detriti rappresentano quindi una minaccia per i satelliti operativi. Pertanto, delle reti di appositi sensori monitorano questa popolazione in continua crescita, che deve essere registrata e catalogata. Questa operazione viene effettuata da sistemi Space Surveillance and Tracking (SST) che permettono di valutare il rischio di collisione in orbita e di caratterizzare le frammentazioni. In questo contesto, le stazioni ottiche di terra permettono di individuare oggetti in diversi regimi orbitali e recuperare informazioni preziose, come la curva di luce del target.
Le curve di luce sono definite come la variazione nel tempo della luminosità del veicolo spaziale. Questi dati racchiudono il moto di tumbling dell'oggetto, il regime orbitale e informazioni legate alla geometria.
Gli approcci nell'analisi di curve di luce possono essere computazionalmente costosi, senza riuscire a collegare un oggetto a una piattaforma. Si utilizzano infatti modelli assai complessi. Queste strategie in particolare richiedono un modello accurato della geometria analizzata e la formulazione più semplice è ottimizzata per gli oggetti convessi. Se si contemplano altre geometrie, il riconoscimento fallisce del tutto.
Altre strategie possono quindi tornare utili, per esempio il "machine learning". Le capacità dei Neural Network (NN) sono state dimostrate in molti settori - compreso quello spaziale -, per tutti i tipi di applicazioni, come l'elaborazione delle immagini o il riconoscimento vocale. Questi sistemi sono in grado di trovare una relazione tra ingresso e uscita senza una formulazione esplicita. Essi "imparano" caratteristiche attraverso il "training" del modello.
Grazie a questa vasta gamma di applicazioni, è reso possibile non solo l'identificazione diretta dell'oggetto, ma anche mettere in relazione una curva di luce sconosciuta con un oggetto noto.

In questa tesi, lo strumento M**a**chine Lea**r**ning based L**i**ght curv**e** Ana**l**ysis (ARIEL) è presentato e declinato in tre sotto-reti:

- **R**ocket b**o**dies Li**g**ht c**u**rves Id**e**ntification (ROGUE) si occupa del l'identificazione di una categoria di oggetto
- **L**ight curve **I**de**n**tification an**d** Corr**e**latio**n** (LINDEN) si focalizza sull'estrazione di caratteristiche e la successiva valutazione della correlazione tra due curve di luce
- **S**iamese Network L**i**ght curv**e**s Co**rr**el**a**tion (SIERRA) effettua una valutazione della somiglianza tra due curve di luce

Queste reti utilizzano una combinazione di Convolutional Neural Network (CNN) e Long Short-Term Memory (LSTM) per estrarre allo stesso tempo caratteristiche generiche e dipendenti dal tempo dalle curve di luce inserite.
Le curve di luce sono recuperate dal database pubblico mantenuto dall'osservatorio Mini-MegaTORTORA (MMT-9). Successivamente, una fase di pre-elaborazione riduce il rumore utilizzando il filtro Savitzky-Golay (SavGol).
I tre modelli vengono poi ampiamente addestrati e valutati in termini di capacità di addestramento e precisione delle previsioni variando i set di dati.
Le prestazioni raggiunte confermano che i NNs possono facilmente estrarre caratteristiche e valutare la correlazione tra due oggetti. Tuttavia, questa strategia è limitata dal numero di diverse piattaforme considerate. Infatti, se questo valore è troppo elevato, i tre modelli svolgono a fatica il loro compito.

Parole chiave: curve di luce, reti neurali, correlazione

Acronyms

ADEO Drag Augmentation Deorbiting Subsystem.

ADR Active Debris Removal.

ARIEL M**a**chine Lea**r**ning based L**i**ght curv**e** Ana**l**ysis.

BRDF Bidirectional Reflectance Distribution Functions.

CCD Charged Coupled Device.

CMOS Complementary Metal-Oxide Semiconductor.

CNN Convolutional Neural Network.

DLR Deutsches Zentrum für Luft- und Raumfahrt e.V..

DNN Deep Neural Network.

ESA European Space Agency.

FOV Field of View.

GEO Geostationary Earth Orbit.

HAMR High Area to Mass Ratio.

IADC Inter-Agency Space Debris Coordination Committee.

LCI Light Curve Inversion.

LEO Low-Earth Orbit.

LINDEN **L**ight curve **I**de**n**tification an**d** Corr**e**latio**n**.

LMO Low-Medium Orbit.

LSTM Long Short-Term Memory.

MMT-9 Mini-MegaTORTORA.

NN Neural Network.

RCS Radar Cross Section.

RMS-Prop Root Mean Square Propagation.

RNN Recursive Neural Network.

ROGUE **R**ocket b**o**dies Li**g**ht c**u**rves Id**e**ntification.

SavGol Savitzky-Golay.

SGD Stochastic Gradient Descent.

SIERRA **S**iamese Network L**i**ght curv**e**s Co**rr**el**a**tion.

SST Space Surveillance and Tracking.

UN United Nations.

Contents

1 | Introduction

This introductory part has the aim to contextualise the work done in this dissertation. At first, the problem of space debris is presented, describing why it is of paramount importance to learn more about them. Afterwards, many solutions relating light curves and machine learning to identify space objects are discussed. Eventually the thesis workflow is elucidated.

1.1. The Space Debris Problem

In the past decades, research in the space sector is increasingly moving towards the need for a more sustainable use of the near-Earth environment, in particular, focusing on "space debris". According to the official definition of the Inter-Agency Space Debris Coordination Committee (IADC), "space debris" are "all artificial objects including fragments and elements thereof, in Earth orbit or re-entering the atmosphere, that are non-functional" [1].

In fact, the aforementioned space region is getting overcrowded, even more so in recent times as shown in Figure 1.1.

This phenomenon is amplified as more commercial satellites are making their way in the space environment: due to the fast development of the space economy and the increasing demand of data from space, solutions in the form of miniaturised satellites (i.e CubeSats) or large constellations are more often employed [2]. Moreover, companies and agencies tend to exploit a single launch to bring multiple satellites in orbit: while it certainly reduces the launch cost, on the other hand it further complicates the capability to track each element of the constellation.

This fast growth in commercial satellites only increases the presence of potential debris in the space environment, causing them to outnumber operational spacecrafts: in August 2022 the functional space objects are only over 6000 compared to the 32000 debris tracked and catalogued. However, more than 130 million debris remain unidentified with sizes ranging from 1 mm to more than 10 cm [3].

Even if their sizes may seem insignificant, those debris carry an enormous kinetic energy

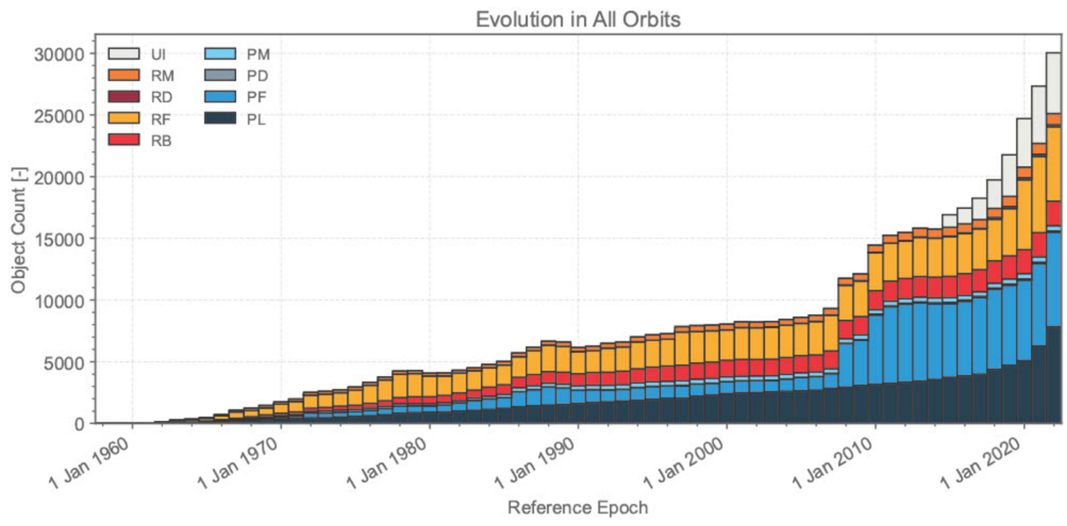


Figure 1.1: Evolution of number of objects in geocentric orbit by object class [2]

and therefore are a threat for operational satellites: an example of an impact with a small debris can be seen in Figure 1.2. The impact is highlighted with a red arrow.

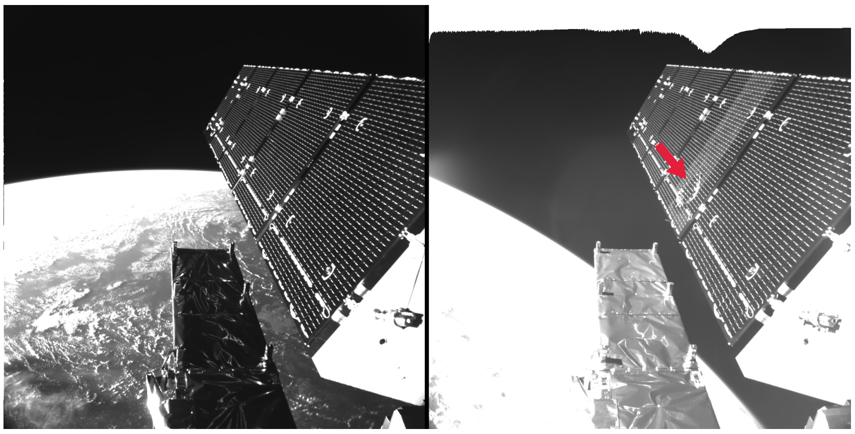


Figure 1.2: Before and after the impact of a millimeter-size debris on the Sentinel-1A solar array [4]

If no countermeasures are taken, as more objects fill the space environment, the probability of impact also increases, leading to an exponential growth of collisions, and therefore of fragmentations which will in turn generate more debris. In the worst case scenario, the space environment will become completely saturated, hence far too hazardous for any space activity. This phenomenon is the so-called Kessler syndrome, named after Donald Kessler who first theorized this problem in 1978 [5].
To avoid this catastrophe, many agencies invested in possible solutions in the framework

of the Space Mitigation Guidelines published in 2002 by the IADC [1]. This document includes recommendations regarding the limitation of space debris released during normal operations, the minimisation of the potential for on-orbit break-ups, post-mission disposal and prevention of on-orbit collisions [2]. Post-mission disposal, for example, deals with United Nations (UN) regulations on re-entry and protected Low-Earth Orbit (LEO) and Geostationary Earth Orbit (GEO) regions or actual missions of Active Debris Removal (ADR). An example of ADR is ClearSpace-1, depicted in Figure 1.3, a satellite being designed by a Swiss startup in collaboration with the European Space Agency (ESA). This spacecraft will rendez-vous with the debris, capturing it thanks to four robotic arms and de-orbiting it into Earth's atmosphere. The mission is planned for 2026 and the first target selected is a 112 kg upper rocket stage in orbit since 2013 [6].
Another interesting way for dealing with the end-of-life is to de-orbit the satellite by increasing its aerodynamic drag. This is done by exploiting a subsystem containing a large membrane supported by four booms: this membrane is then unfolded at the end of the satellite mission and will act as a sail, accomplishing the desired effect. The deployed configuration is shown in the artist impression in Figure 1.4. However this process can last for a very long period of time, up to 25 years. The prototype is called ADEO and is currently being designed by DLR [7].

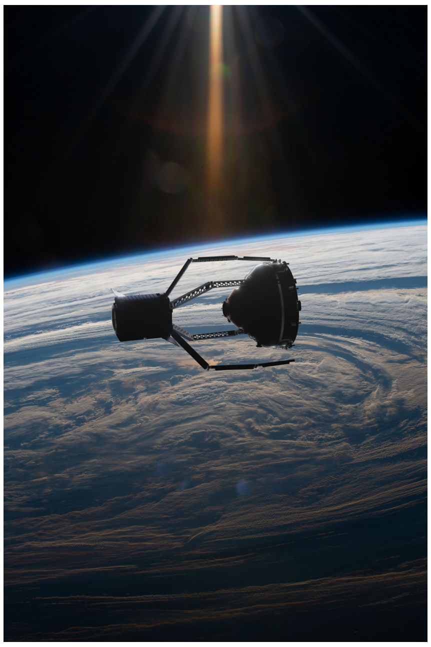


Figure 1.3: ClearSpace-1 [6]

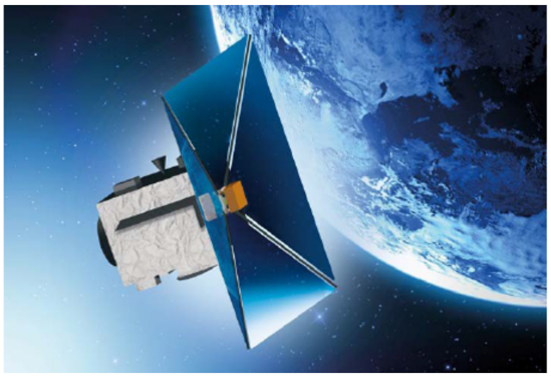


Figure 1.4: ADEO [7]

All those problems convene on the need to identify space objects directly from observations. This is one of the operations done by the Space Surveillance and Tracking (SST) systems: a network of ground-based and space-based sensors which survey, track, catalogue and characterise space objects. The aim is to assess the risk of in-orbit collisions or of un-controlled re-entries and detect and characterise in-orbit fragmentations [8]. The obtained data are then used for orbit determination and characterisation of the object and to update the catalogue of the man-made space objects. A summary of the activities done in SST are summarized in Figure 1.5.

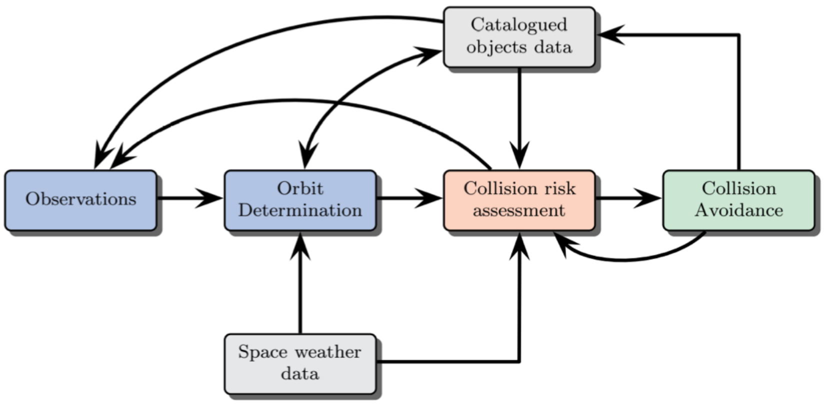


Figure 1.5: SST Operations [9]

The fundamental part of the SST activities are the observations.
Different types of observations can be carried out from ground-based to space-based sensors, using passive or active optical sensors or radars [10].
Considering space-borne sensors, observations are obtained exposing satellites in the space environment so as to model the debris population. This method allows to detect targets smaller than 1 mm, much smaller than what ground observations can accomplish.
The surveys done in the SST framework are all ground-based and the systems employed are detailed hereafter.

Passive optical observations focus on collecting sunlight reflected by the object surface. From there, angle measurements and astrometric data are retrieved and then employed for orbit determination and object cataloguing. However, this type of observation requires that: 1) the object is illuminated by the Sun; 2) the observations are performed at night-time; 3) the weather conditions must be suitable.

Active optical or Satellite Laser Ranging (SLR) systems are still based on the reflection of the target, from which range measurements are obtained. However, in this case, lasers are employed to induce the reflection. They are less restrictive than passive, however they still necessitate good weather conditions.

Radar systems can acquire many types of information, i.e. the Doppler shift between received and transmitted frequency, angular positions (azimuth and elevation), the received power, the range of target and polarization changes, without the limitations of the optical systems. These data can also lead to the evaluation of the Radar Cross Section (RCS).

Focusing on the first class, the objects' light curves can be retrieved thanks to photometry. Light curves describe the variation in time of the magnitude of the object of interest (which can be natural, like stars, or artificial like in this case), and therefore they inherently contain all the pieces of information necessary for the identification of the object and its attitude motion. In Section 2.1 further explanations regarding this powerful property are elucidated.
Oftentimes, object geometry can be reconstructed from light curves using analytical methods called Light Curve Inversion (LCI). However in many cases this leads to complex formulations and time-consuming operations. Therefore the trend in the past years is to employ machine learning to recognize space objects features and thus allow the identification.

1.2. State of the art

Light curves are an incredible tool for the characterisation of space objects. In fact, they encapsulate information on both motion and geometry of the object.
Thus far, LCI techniques are employed and have proved to be effective for asteroid shape recognition. A first attempt of using LCI for artificial space objects is proposed by Bradley and Axelrad in 2014 [11]. Their idea is to estimate the shape of GEO objects by simulating light curves taken from a space-based sensor in Sun-synchronous orbit and then feed them to a LCI algorithm. However the technique they employed focuses on convex shapes: as a result, it faithfully reconstructs naturally convex items like upper stage rockets or CubeSats while having some difficulties for High Area to Mass Ratio (HAMR) objects and box-wing satellites, as shown in Figure 1.6.
The model used for the LCI is in fact far too simplistic, therefore more complex models like the Phong Reflection Model, a subcategory of Bidirectional Reflectance Distribution Functions (BRDF) [12], need to be used. This innovative model is applied for example by Furfaro and Linares in [13] and [14] and by McNally et al in [15].
In any case, analytical models based on estimation theory are rarely employed in the strategies above-mentioned, as machine learning provides accurate shape identification efficiently, in particular in terms of computational capabilities.
Since deep learning is proved to be an extremely powerful tool for classification problems,

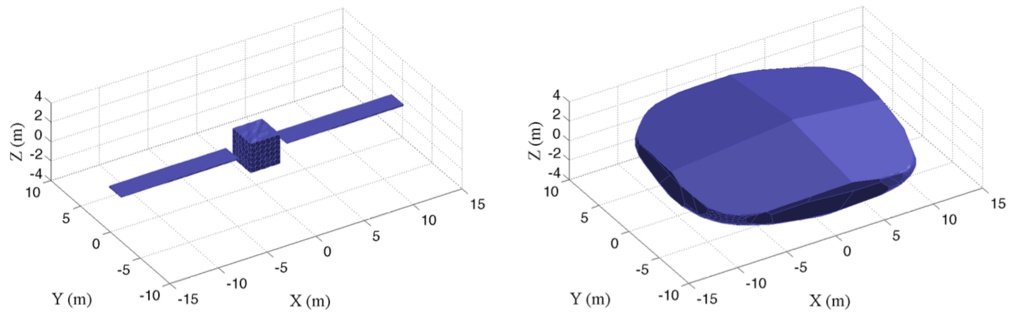


Figure 1.6: True and LCI shape models for box-wing satellite in GEO [11]

they may be suitable for applications in the space domain, in particular for the identification of space objects. Linares and Furfaro in [13] were one of the first to try to move in this direction. They built a Convolutional Neural Network (CNN) which classifies simulated light curves into three classes: debris, rocket bodies and payload. Those objects are modelled combining simple geometries. Then the light curves are obtained according to the above-mentioned Phong Model [12], considering spacecrafts in GEO orbit with the same attitude regime, as can be seen in Figure 1.7.

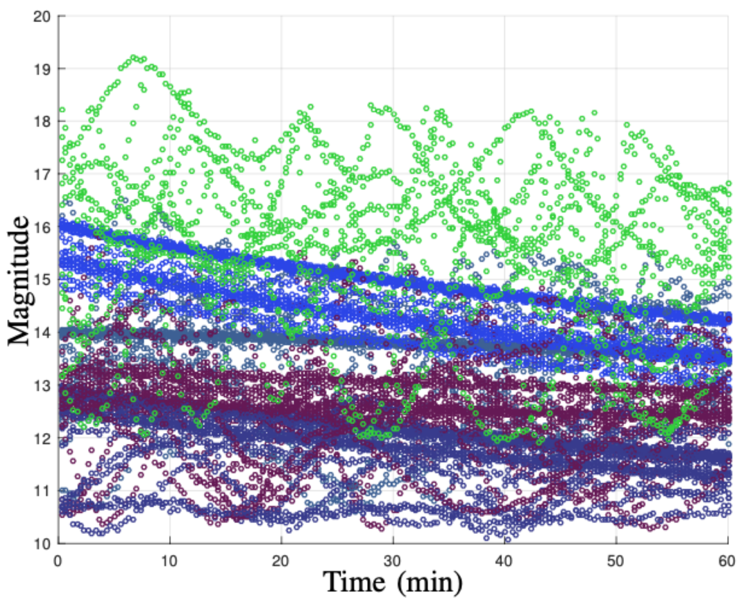
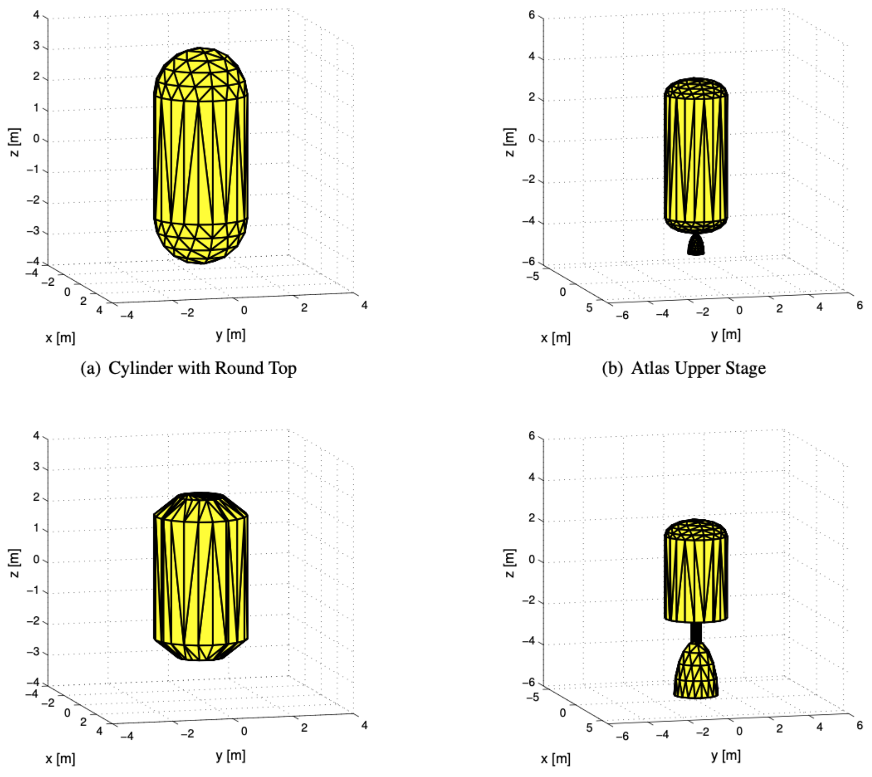


Figure 1.7: Example of simulated light curves [13]

5000 scenarios are tested, resulting in a 99.6% accuracy on correct class prediction. CNN therefore can provide an extremely accurate classification with a simple and computationally efficient implementation.

It is no surprise that they are also employed for shape identification, as proposed by Furfaro, Linares and Reddy [14]. In this case, the aim is to employ CNN to determine the shape of second stages of rocket bodies. The shapes considered are shown in Figure 1.8.



(a) Cylinder with Round Top (b) Atlas Upper Stage (c) Cylinder (d) Falcon 9 Upper Stage

Figure 1.8: Representative shapes for rocket bodies [14]

In this case the overall accuracy reached 92%.
Taking these findings as a starting point, McNally et al tries to identify satellites, considering six different platforms in GEO with different attitude modes: two tumbling and four stable [15]. Light curves are still simulated, however a validation process is included to ensure the closeness with real data. This is performed to avoid the so-called "domain gap", as the performances are vastly different considering simulated with respect to real data.
Three different approaches are then analysed: 1) using only raw data, with no input processing; 2) carrying out a feature analysis to input only the characteristics that are likely influencing the light curve; 3) realizing a differential process to include also the variation in time of the brightness instead of absolute values. The three approaches do not consider CNN but Fully-Connected Neural Networks.
The first and third approach reach over 90% accuracy even though the data volume in the latter is over 4 times the former's. The second approach performed well especially on tumbling satellites, where the accuracy is greater than 95%. It is worth mentioning

that the testing of the raw data approach on real light curves ensured a 70% accuracy in differentiating two stable platforms. This small decrease in the performances was mainly due to the above-mentioned "domain gap": in fact the real data were processed in a slightly different way than the simulated case, therefore the network has to re-adapt itself to counteract these differences.

Light curves are by definition time-varying, therefore Recursive Neural Network (RNN), employing Long Short-Term Memory (LSTM) models, can be exploited, due to their capability of extracting features from sequence-dependent data. An example is proposed by Kerr et al [16]. Here, light curves belonging to objects in the GEO region are pre-processed into an Auto-encoder to independently extract features which are consequently fed into two different networks: one consisting of CNN and another of LSTM cells. The dataset is composed by both simulated and real data, divided in three shape classes: sphere, box and box with wings. As a result, taking into account that the LSTM requires more training - as mentioned in the article -, both models can properly distinguish those type of shapes. The CNN network even reaches 100% accuracy for the sphere and box with wings cases, while the LSTM one still achieves over 85% accuracy in the three cases.

1.3. Proposed workflow

Provided the considerations discussed in the previous Section, NN have been proved to be proficient in directly identifying shapes from light curves and therefore classify objects, in terms of type or even platforms in some cases.
Some points however need to be taken into account:

→ All the objects analysed are in GEO orbit and have a defined attitude state: this leads to a simplified classification problem as the problem is exclusively related to the geometry considered.

→ Many strategies employ synthetic light curves in order to increase the data volume or for ease of use: even though the synthetic light curves are pretty realistic, some aspects may not be simulated correctly, like, for example, noise. This may lead to the above-mentioned "domain-gap".

In addition, a step further can be done if, on top of a classification, it is possible to evaluate the correlation between an unidentified object with respect to a known one.
In this dissertation, an attempt on using real light curves to assess a degree of correlation between two space objects is performed. The light curves analysed have been recovered

from the Mini-MegaTORTORA (MMT-9) database and focus on objects in the LEO region presenting a periodic or aperiodic attitude motion.
As part of the M**a**chine Lea**r**ning based L**i**ght curv**e** Ana**l**ysis (ARIEL) architecture, three strategies are devised, each one having a distinct objective:

- ROGUE involves the direct recognition of the rocket body type among the light curves given;
- LINDEN introduces multi-class classification to determine if two objects belong to the same category or not;
- SIERRA introduces a peculiar network structure, the Siamese Network, to evaluate the distance between predictions made on the type of object observed.

The strategy employed in SIERRA employs a different structure than typical NN. The Siamese Network, further described in Section 2.2, compares two inputs in relation to a reference one, in order to determine their similarities. Therefore, it is possible to evaluate how close an initial prediction the characterisation of the observed object is. Additionally, it can act as a validation process for LINDEN.

All the strategies mentioned involve Neural Networks, in particular CNN and RNN. The performances of each network is then assessed in terms of loss and defined metrics, e.g. accuracy for ROGUE and LINDEN and similarity for SIERRA.
This dissertation structure is listed hereafter:

Chapter 2 outlines the theoretical background needed on light curves and deep learning

Chapter 3 describes the data retrieval and pre-processing to prepare the light curves for the NN

Chapter 4 defines the ARIEL's architectures employed, where each strategy has the aim to achieve a specific task

Chapter 5 presents the results obtained and comments the outcomes

Chapter 6 draws the conclusions and proposes possible future developments

Figure 1.9 summarizes the above-mentioned steps.

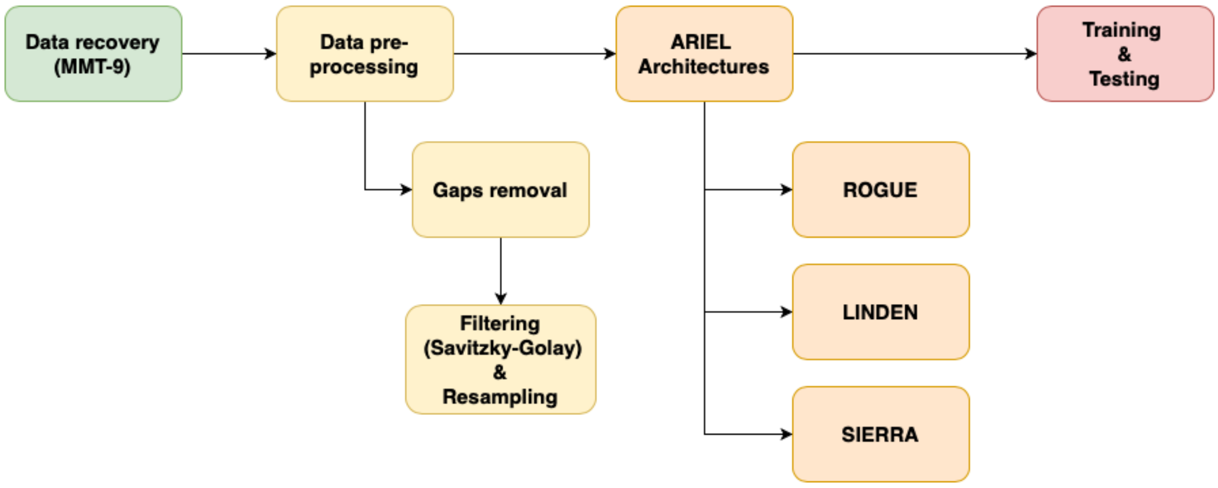


Figure 1.9: Thesis workflow

2 | Theoretical background

The aim of this chapter is to provide the theoretical background for the two main concepts analysed in this dissertation: light curves and machine learning.
Starting from light curves, a preliminary introduction on optical telescopes and detectors is carried out. Then the procedure on how to obtain light curves is tackled: the particular case of the Mini-MegaTORTORA (MMT-9) observatory is studied, as it is the main resource of light curves in this dissertation. Eventually, a brief presentation on the the Savitzky-Golay filter is performed as it is extensively used for the light curve preprocessing.
Subsequently, machine learning fundamentals cover the definition and structure of a Neural Network, its training working principle and the main criticalities that can be encountered, in particular over- and underfitting. Focus is put on CNN, RNN and Siamese networks, since they are the main architectures employed in ARIEL.

2.1. Light curves

Optical telescopes

Telescopes are still the main tools used to perform observations of space and to retrieve and catalogue light curves of natural or man-made space objects. These instruments can be both on ground or on orbit and their type is named after the part of the spectrum they cover.
Ground telescopes consider only visible ($\lambda = [400; 700]$ nm) or radio ($\lambda = [1 \text{ mm}; 20 \text{ m}]$) spectra, since the Earth atmosphere is opaque to radiations outside those ranges. This phenomenon is highlighted in Figure 2.1.
As light curves are obtained from observations in the visible range, optical telescopes and their challenges are further analysed.
Granting that the atmosphere is not opaque to visible radiations, some distortions in this range, however still occur: three phenomenons take place and have to be taken into account during the survey. They are:

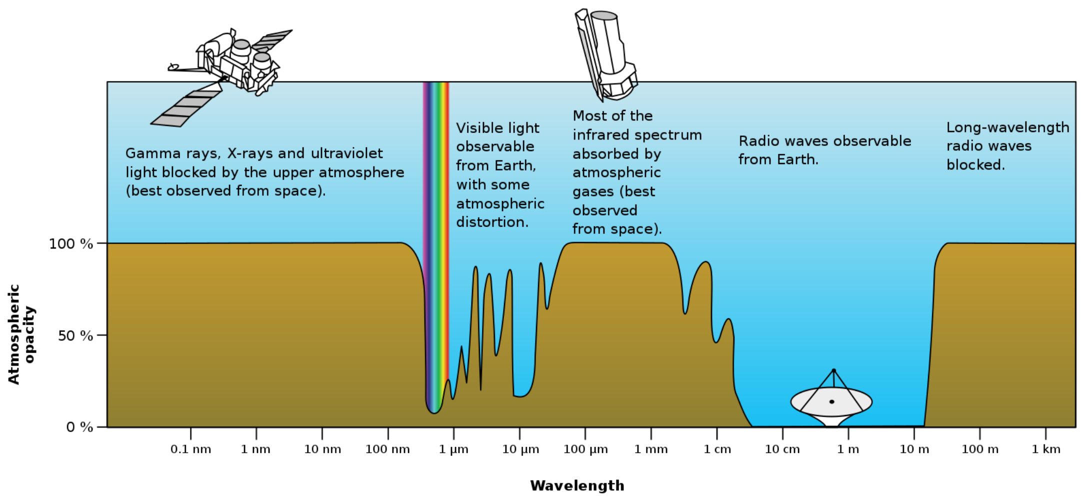


Figure 2.1: Atmospheric opacity [17]

Scintillations : The wavefront is deformed due to the air not being steady since the atmosphere has different layers with varying densities and temperatures. Therefore the amount of light reaching the observer is different, thus the light source is seen to "scintillate"

Seeing : The light is refracted in different ways along its path across the atmosphere: the image received is smeared and point sources can be seen as vibrating speckles

Extinction : This phenomenon combines scattering and absorption of the incoming light by atmospheric molecules and dust

Moreover, good observations of very faint objects optical telescopes require darkness and transparency, in particular dry air, clear sky and good seeing: this explains why optical telescopes can be found in desert areas or on top of mountains.
The main objective of optical telescopes is in fact to collect light from a large portion of the sky so that it is possible to study even very faint sources with a very high resolution, and to measure the position of the observed object.
Two types of telescopes are employed: refractors use lenses while reflectors employ mirrors. Their structure is consequently different as can be seen in Figure 2.2.

Refractors

Refractors are composed by two lenses named the "objective" and the "eye-piece". The objective collects the incoming light and focuses it onto the focal plane, while the eye-piece behaves like a small-magnifying glass and allows the observer to look at the recomposed image. The distance between the eye-piece and the focal plane can be adjusted in order to improve the focus of the image.

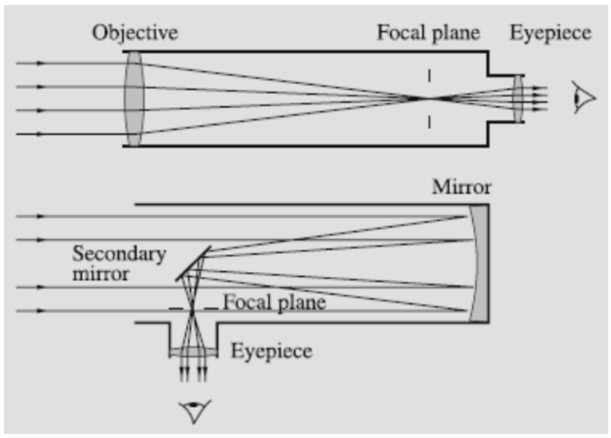


Figure 2.2: Refractor (top) VS Reflector (bottom) telescopes [18]

On top of a direct observation of a space object, the telescope can be connected to a detector which processes the reconstructed image and convert it into an electric signal that can be collected and analysed.

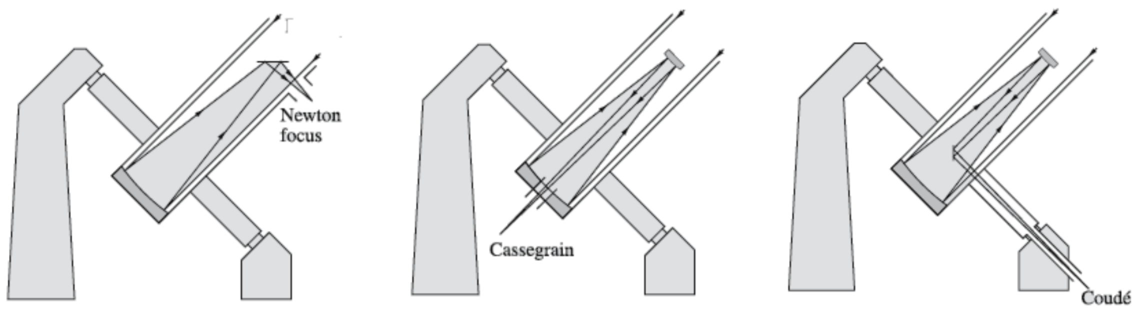
The main limitation for this type of telescopes is the chromatic aberration. This problem occurs because the refraction coefficients depends on the wavelength λ: as a consequence, as the incoming white light is decomposed by the objective, different focal planes are formed for each component of the light. To solve this issue, an achromatic doublet can be used: it is formed by two lenses with opposite refractive index that cancels out the aberration.

The use of this type of telescope is limited due to its restricted FOV and long structure. However it is still employed for the observations of celestial bodies such as binary stars or for measuring the position of stars.

Reflectors

Reflectors are the most common optical telescopes. The working principle is as follows: light is collected by a parabolic mirror coated with a thin aluminum layer and then focused in a unique focal point. As lenses are not involved, the problem of chromatic aberration is avoided. The created image is then observed through an eye-piece or guided to a detector. Different configurations can be used as depicted in Figure 2.3, each having their own peculiarities.

The simpler is the Newton configuration, where the light exits perpendicular to the incoming direction, through the so-called Newton focus. Then, the Cassegrain design considers two main mirrors, the primary and the secondary: the primary is parabolic and has a hole in the center, in such a manner that the light is first reflected onto the hyperbolic secondary mirror at the front and finally focused into the Cassegrain focus. This configuration is suitable for spectrography and photometry as instruments can be mounted onto



(a) Newton (b) Cassegrain (c) Coudé

Figure 2.3: Types of reflector telescopes [18]

the secondary focus. Last but not least, the Coudé architecture exploits multiple reflections such that the light is guided through the declination axis and into the Fixed-Coudé focus. Its main application is accurate spectroscopy, however it is worthy to be mentioned that the light that reaches the detector is around 30% of the original amount due to the many reflections it encounters.

Similarly to their refractor counterparts, the reflector telescopes suffer from a type of aberration: the coma. In fact, if the images are displaced from the optical axis, rays don't converge in a single point and form a comet-like shape. This aberration leads to having a small FOV, however this defect can be counteracted with the use of correcting lenses.

Two parameters allow to evaluate the capabilities of telescopes: the resolving power and the aperture ratio.

The aperture ratio F is defined by the ratio between aperture D - the diameter of the objective - and focal length f - the distance between the objective and the focal plane -. In general, it is denoted as f/n - where n is a positive integer - and characterises the light-gathering power of the telescope: the higher F is, the "faster" the telescope is. This means that photographs can be taken with short exposures, since the image is bright. Typical values for the above-mentioned telescopes are summarized in Table 2.1.

Telescopes	**Refractors**	**Reflectors**		
		Newton	**Cassegrain**	**Coudé**
Aperture ratios f/n	$f/10$, $f/20$	$f/3 \div f/10$	$f/8 \div f/15$	$f/30 \div f/40$

Table 2.1: Typical values of aperture ratios [18]

The other fundamental parameter is the resolving power, defined as the minimum angle of separation between two objects. The theoretical limit is imposed by the diffraction: if two sources are too near, they are observed as a single oblate disk. According to Rayleigh criterion, the resolution θ is linked to the aperture D and wavelength λ by the following rule:

$$\sin(\theta) \approx \theta = 1.22\frac{\lambda}{D}$$

Moreover, another limit on the resolution is imposed by the physical sizes of telescopes. In fact, the telescopes may become far too massive to achieve a greater aperture and therefore a greater resolving power.

Detectors

After the incident light is collected, the photons need to be converted into an electric signal to be analysed. Therefore, after collimation, filtration and isolation, the light enters a detector. Generally, these detectors are made of semiconductor material. After the conversion, the signal is digitized, then processed and stored.
The most common detectors found in optical telescopes are Charged Coupled Device (CCD) and Complementary Metal-Oxide Semiconductor (CMOS) cameras, depicted in Figure 2.4. The structure is similar nut they differ in the readout strategy.

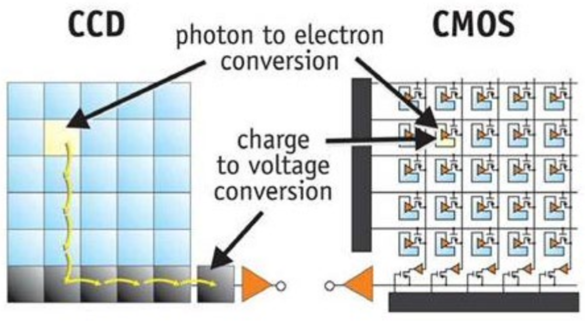


Figure 2.4: CCD vs CMOS cameras [19]

These cameras are made of a surface with silicon photodiodes in arrays of pixels. When a photon impacts a pixel, it releases an electron which remains trapped. In a CCD camera, potential differences are scanned in order to move the accumulated charges row-by-row into a readout buffer. Instead, in a CMOS camera, each pixel has its own readout capacity, therefore they can be isolated without impacting the adjacent ones.
The detector is linear in both configurations as the number of electrons trapped are proportional to the number of impacting photons.
CCD cameras are much simpler with respect to CMOS, therefore they provide lower-noise and higher quality measurements. However, due to their readout strategy, the CCD reading speed is much lower; hence, for increasing resolutions, they require a mechanical

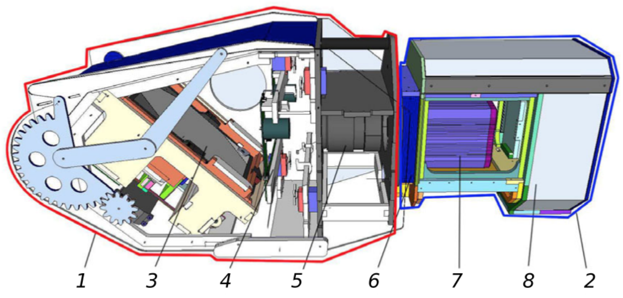
shutter to halt the measurement process and to avoid contamination.

The Mini-MegaTORTORA system

The light curves used in this dissertation are recovered from the Mini-MegaTORTORA (MMT-9) observatory [20][21]. This telescope was developed jointly by Kazan Federal University (RUS), Special Astrophysical Observatory of the Russian Academy of Sciences and Parallax Ltd and started its operations in 2014. Its aim is to study optical transients, therefore not only man-made objects but also meteors or events like gamma-ray bursts. Hereafter are outlined the characteristics of MMT-9 [20]:

- Nine-channel optical wide-field monitoring system
- FOV = 900 deg^2
- Temporal resolution = 0.1 s
- Angular resolution = 16" per pixel
- Detection limit ≈ 10 mag
- Capable of detecting from 200 to 500 tracks every night, from every orbital regime
- Accuracy of coordinate determination ≈ 5-30"

The telescope is divided in nine channels, one of which can be visualised in Figure 2.5a, each mounted in pairs of five equatorial mountings, therefore mounted in the Polar-Declination plane. Figure 2.5b gives a general view of the system.



(a) Channel of MMT-9 [21]

(b) General view of the system [21]

Figure 2.5: MMT-9 system overview

Referring to Figure 2.5a, each channel consists of:

1,2: Protective case

3: Coelostat mirror which can be rotated by 10° about two axis, granting ±20° FOV adjustment

4: Set of color and polarization filters which can be introduced during the observations

5: Canon EF 85 $f/1.2$ objective

6: Optical corrector

7: Andor Neo sCMOS detector, characterised by 30 Hz readout frequency (here, 10 Hz is employed), quite high quantum efficiency and low readout noise. The detector has a build-in cooling system to keep it at constant temperature of -40°C

8: Heating and ventilation system

The mounting is then installed in a automated housing, containing also the power supply for the actuators driving the mechanical elements of the observatory. The system is also equipped with weather sensors, a very-wide field camera for all-sky surveys and a night-sky brightness sensor.
The combination of the movement of the mirror and the set of filters allows the telescope to operate in two ways, as shown in Figure 2.6. First of all, a clear wide-field monitoring is performed which detects a transient, thanks to its overall 900 deg^2 FOV. Then, color and polarization filters are engaged and narrow-field observations are made considering 100 deg^2 FOV. During the narrow-field observations, all the objectives point the transient previously detected and record it employing different combinations of the filters installed.

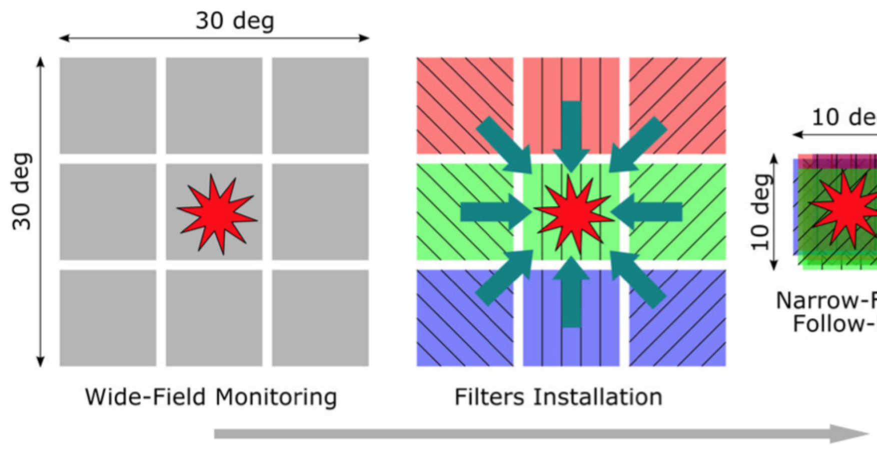


Figure 2.6: Operation modes of MMT-9 system [21]

All the facility is controlled by a dedicated software which takes care of different oper-

ations. An example is the sky wide-monitoring: it is governed by an automatic survey scheduler which selects the optimal portion of sky to observe.
Another operation done by the software is real-time data processing which engages the narrow-field follow up once a transient is detected. Moreover, a control software sensible to external triggers that can come from other telescopes (space or ground based), can modify the observation mode.
The software is divided among twelve computers: one for each channel to allow data acquisition, real-time analysis and storage; one spare maintained in standby; two independent ones for the control of the overall system. The observatory is completely automated, managing scheduling, high-level data analysis and storage.
Two databases are also handled by the software. The first contains astronomical objects - asteroids or transient events like gamma-ray bursts - while the other comprises of man-made space objects, which is of interest for this dissertation. The information that can be found in that database are described in Section 3.1.

Light curves

As mentioned in Section 1.2, light curves express the variation in time of the brightness of an observed object, which can be either natural, i.e. asteroids or stars, or artificial, i.e space debris. This dissertation focuses of the latter case of artificial debris.
The brightness of the object can be expressed in terms of its emitted power, so-called luminosity, which is actually declined in two different quantities:

- Intrinsic luminosity L which comes effectively from the source
- Apparent luminosity l which depends on what it is effectively observed, therefore on the distance observer-object, the intrinsic luminosity of the object and if extinction occurs along the light's path to the observer.

Intrinsic luminosity L can be retrieved considering a spherical surface emitting a flux l with radius posed as the distance d between the source and observer:

$$L = 4\pi d^2 l \tag{2.1}$$

However, luminosity is in general quantified in terms of magnitudes. Apparent magnitude m is linked to apparent luminosity in a logarithmic way:

$$m_2 - m_1 = -2.5 \log\left(\frac{l_1}{l_2}\right) \tag{2.2}$$

where 1 and 2 represent two distinct objects with luminosity l_1 and l_2 respectively. It has to be noted that the smaller the magnitude, the brighter the observed object is. Combining Equations 2.1 and 2.2 one can relate distances to apparent magnitudes. In fact, if considering the same object at two different distances, d and d_0, one finds Equation 2.3.

$$L = L_0 \Rightarrow \frac{l}{l_0} = \left(\frac{d_0}{d}\right)^2 = 100^{\left(\frac{m_0 - m}{5}\right)} \tag{2.3}$$

Posing $d_0 = 10$ pc - where pc is parsec and 1 pc = 3.26 ly -, the definition of absolute magnitude M is obtained:

$$M = m + 5 - 5\log d \tag{2.4}$$

Absolute and apparent magnitudes can both be used to obtain the luminosity of the observed object. Given the evolution of these quantities in a defined time-frame, e.g. during a survey, one can finally obtain the light curves. An example of light curve can be shown in Figure 2.7.

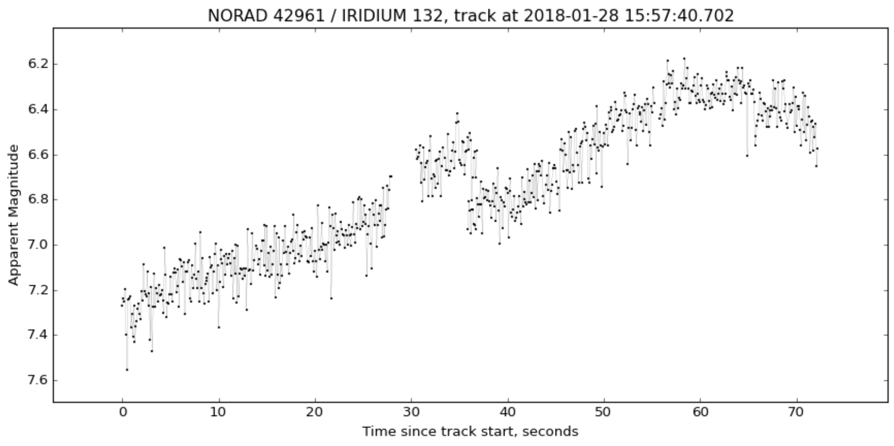


Figure 2.7: Example of light curve [20]

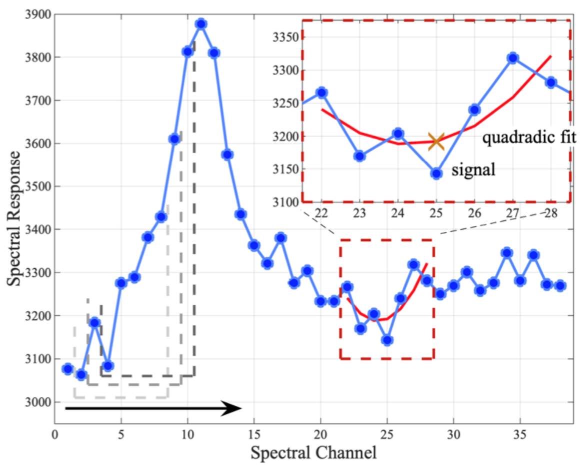
Savitzky-Golay filter

The raw light curves obtained from the MMT-9 database still need to be processed before they can be used in ARIEL, as further elucidated in Chapter 3. In particular, the noise needs to be attenuated to retrieve only the frequency contents of interest. The filter considered is the Savitzky-Golay (SavGol) filter theorized by A. Savitzky and M.J.E. Golay in 1964 [22].
It has two main properties:

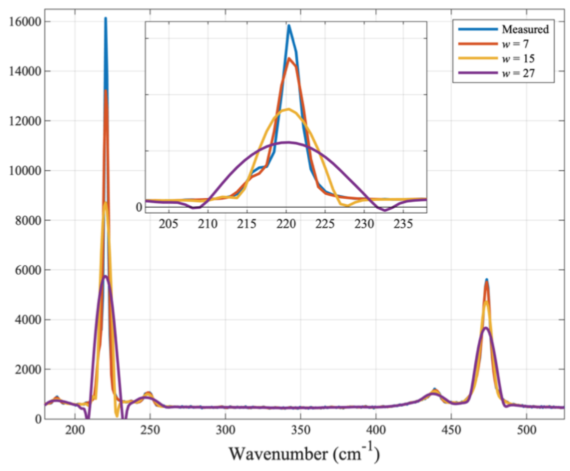
- Smoothing, allowing to reduce high frequency noise

- Differentiation, to limit low frequency signal, e.g due to offsets or slopes

In this dissertation only the smoothing property is actually employed as it was deemed enough for the pre-processing of the data. The working principle is simple [23]: given a signal with N points, a window of width w is considered and it is moved across the signal. In each window, SavGol computes a polynomial regression of order o: the estimate is given at the center point of the window, hence w is typically an odd integer number. Moreover an additional restriction is added to operate correctly, in fact the number of measurements must be greater or equal than the number of parameters to be estimated, therefore $w \geq o + 1$. To benefit from the smoothing effect however $w > o + 1$.
Figure 2.8a shows how the SavGol filter operates while Figure 2.8b gives an impression on the smoothing properties: in fact, it can be noted that, with fixed o, as w increases, the smoothing increases and the peaks are more suppressed. Therefore a proper tuning of the parameters (w, o) needs to be performed to remove noise without removing the main signal content.



(a) Procedure for SavGol ($w = 7, o = 2$)



(b) Smoothing effect of SavGol

Figure 2.8: Savitzky-Golay filter working principle and effects on a signal [23]

2.2. Deep learning fundamentals

Neural Networks (NN) have been extensively used in many applications, such as computer vision or speech recognition, even inside the space sector as show the examples discussed in Section 1.2.
NN basically function like a biological brain - hence "neural" -: these systems have in fact the ability to "learn" and improve from experience without any prior explicit program-

ming. The "learning" process has the goal to search for patterns in data or observations to improve the prediction done at the output.
Three types of learning processes can therefore be defined:

Supervised : Every input data is related to an output label, therefore the network searches for an input-output relationship.

Unsupervised : Input samples are not related to a label, thus the network groups similar patterns into clusters.

Reinforcement : As a complex objective needs to be achieved, the NN has to adapt itself in such a way to maximize its performances. This process employs agents which are "rewarded", using a so-called reinforcement signal, in order to learn which action is best.

As ARIEL is based on Supervised learning, this case is the one studied in depth.

Structure of a Neural Network

The structure of a NN, depicted in Figure 2.9, is as follows: between the input and the output layers, one or many hidden layers unfold, each containing a fixed number of nodes which are connected to the ones in the previous and the subsequent layer. The nodes are called "neurons" due to the above-mentioned resemblance with the brain.

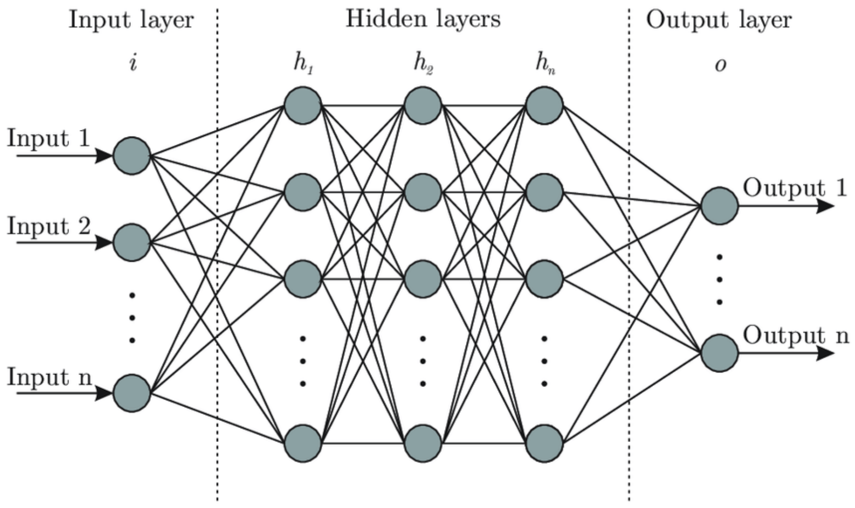


Figure 2.9: Architecture of a simple NN [24]

Each neuron executes the following operations:

1. It takes an input **x**, from the input layer or from the previous hidden layer

2. The input is linearly combined through a weight matrix $\mathbf{W}$ and a bias vector $\mathbf{b}$ such that: $\mathbf{z} = \mathbf{W}\mathbf{x} + \mathbf{b}$
3. The result of the linear combination goes through an activation function $f(\cdot)$ which maps the result into a defined range - depending on the function used - and "activates" the neuron if a condition is met: $\mathbf{a} = f(\mathbf{z})$
4. The output $\mathbf{a}$ is then fed to the next hidden layer or displayed in the output layer

These operations are iterated for each neuron in each hidden layer until the output layer is reached. For the sake of clarity, a neuron in shown in Figure 2.10.

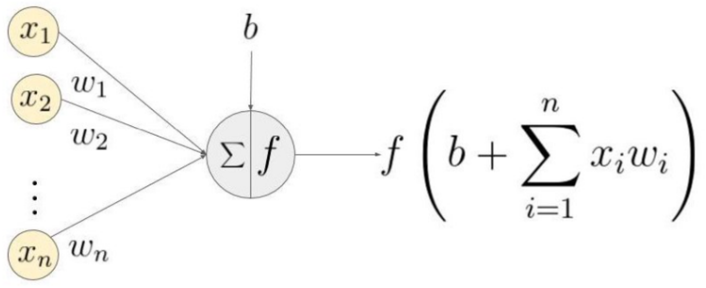


Figure 2.10: Neuron structure [25]

As mentioned, each neuron is characterised by incoming weights $\mathbf{W}$ and a bias $\mathbf{b}$. The weight summarizes the strength of the connection the neuron receives therefore the higher it is, the more significant the previous neuron is. Given all the incoming connections from the previous layer, a weighted sum of the input is computed. The bias is an additional term to grant a threshold so that the activation of the neuron is meaningful.
Then this linear combination is passed to an activation function which can have various formulations, some of them are shown in Figure 2.11. As the name states, these functions "activate" the neuron if a certain condition is met and the neuron can transmit its contribution to each neuron of the next layer.
The simplest structure that can be achieved consists of a single hidden layer: in this case it is called Fully-Connected Neural Network. If multiple hidden layers are stacked, the network is considered a Deep Neural Network (DNN). CNN and RNN, described afterwards, are special types of NN and have peculiar layers on top of the Fully-Connected ones.

Peculiar network structures

Different types of NN can be considered according to the problem at hand. The most relevant are summarized hereafter:

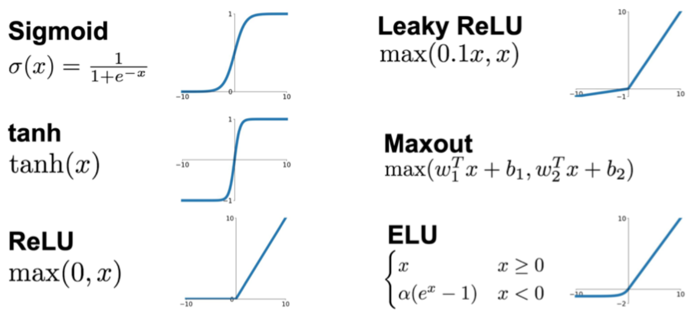


Figure 2.11: Examples of activation functions [25]

Fully-Connected Neural Networks : The simplest NN, composed of input, hidden and output layers which tries to retrieve a data-label relationship.

Convolutional Neural Networks : Particularly indicated for computer vision applications, this type of NN combines a set of operations such as convolution and pooling, which reduce the input in such a way that it becomes easier to process while retaining the most important features for a good prediction.

Recursive Neural Networks : When sequence data is involved, i.e language processing, this NN is recommended due to the fact that it takes into account the order by which the input enters the network.

The ARIEL networks comprises both CNN and RNN, in particular of one-dimensional CNN and Long Short-Term Memory (LSTM) cells.
The explanations done for the CNN are made for 2D input data, typically images, but the working principle applies also to 1D data. Regarding RNN, other structures are available, i.e Gated-Recurrent Unit (GRU) or Bidirectional RNN, but in this dissertation the focus will remain on LSTM cells only.

Convolutional Neural Networks (CNN)

This type of NN are composed mainly of three types of layers: in addition to Fully-connected layers already described, Convolution and Pooling layers are added.

Convolution layers

Convolution layers - as the name states - perform a convolution of the input matrix with a weight matrix called filter or kernel. This particular matrix is in fact slid through the input image with a particular stride: given a 2D input matrix of dimension $n \times n$ and by applying a $k \times k$ kernel every s steps, the resulting matrix is of dimensions $\left(\frac{n-k}{s} + 1\right) \times \left(\frac{n-k}{s} + 1\right)$.
The convolution operation therefore allows the network to retain the main features from the input matrix, i.e the position of the edges or the color value if the input is a RGB

image.
An example of this operation can be seen in Figure 2.12. The convolution considers the dark blue sub-matrix of the input and the orange matrix - the kernel -, the result being the dark green element.

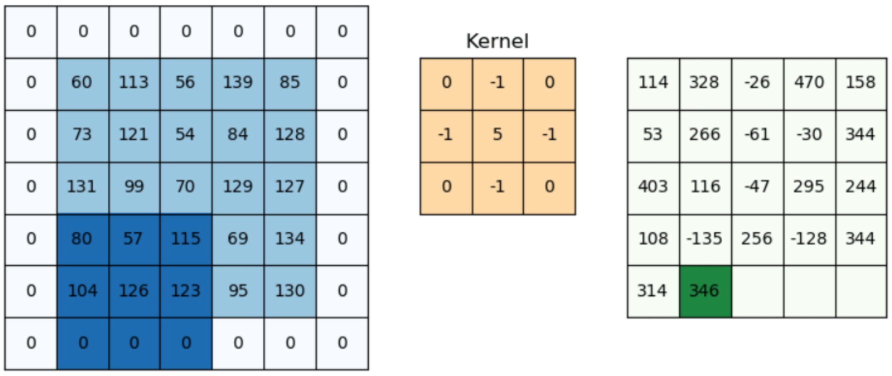


Figure 2.12: Example of a convolution operation [26]

In Figure 2.12 another peculiarity can be observed: in the blue input matrix, a frame of zeros is put in place. This procedure is called padding and allows for each pixel to be used evenly: in fact, if not applied, pixels at the corners are used much less than the center ones, and a lot of information may be lost. The output matrix, considering also padding p and stride $s = 1$, is of the same dimensions of the input matrix, therefore $p = \frac{k-1}{2}$. As a consequence, k is generally considered odd, even more so to guarantee that the center value of the kernel has a central position.

Pooling layers

Pooling layers often follow Convolutional ones and are employed to reduce the size of the data, consequently reducing the computation time of the CNN.
The input matrix is divided in submatrices according to a defined filter size and stride. Then the average or the maximum value is taken from each submatrix. The working principle is shown in Figure 2.13.

Recursive Neural Networks (RNN)

As stated previously, RNN are particularly indicated for sequence data inquiry, i.e natural language processing, forecasting applications or sentiment analysis. In fact, the sequence in which data is given is an additional feature that can not be discarded.
RNN includes a recurrent unit that memorizes the previous state computed: the neuron at time $< t >$ therefore receives the previous neuron output at time $< t >$ and the output from the recurrent unit at time $< t - 1 >$. Figure 2.14 gives a simple representation of a

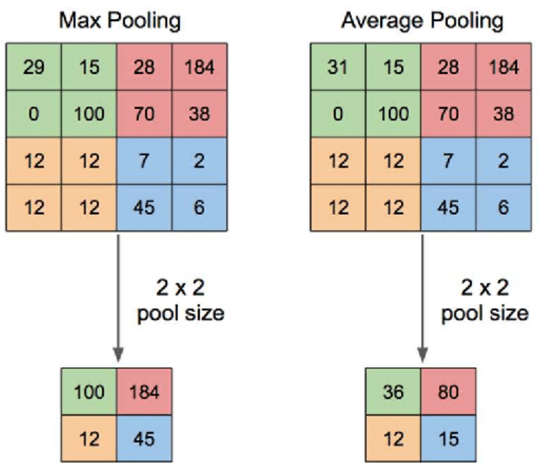


Figure 2.13: Pooling operations: Max Pooling and Average Pooling [27]

single layer RNN. By extending this case, it is possible to obtain multiple recurrent layers.

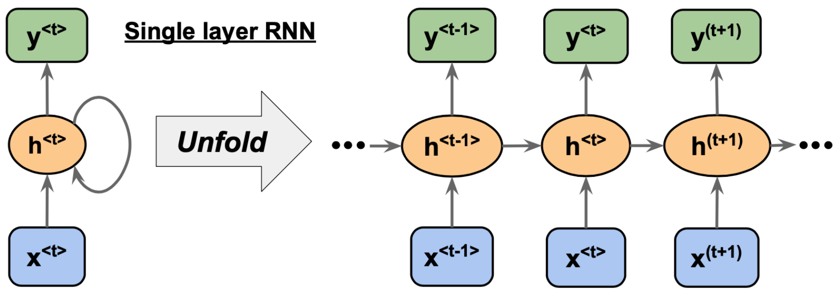


Figure 2.14: Single layer RNN working principle [28]

Different configurations can be used, depicted in Figure 2.15 and summarized in Table 2.2.

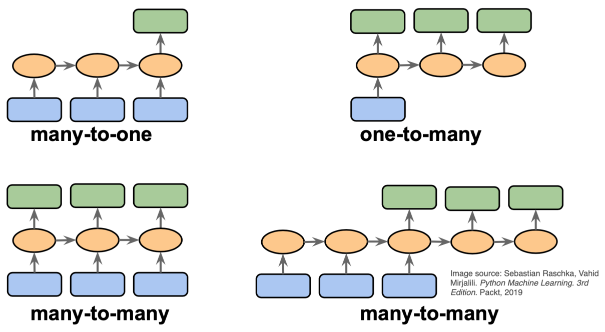

many-to-one
one-to-many
many-to-many
Image source: Sebastian Raschka, Vahid Mirjalili. Python Machine Learning. 3rd Edition. Packt, 2019
many-to-many

Figure 2.15: Possible RNN configurations [28]

Configurations	Is input a sequence?	Is output a sequence?
Many-to-one	✓	×
One-to-many	×	✓
Many-to-many (direct or delayed)	✓	✓

Table 2.2: Solutions found given the initial condition

Long Short Term Memory (LSTM)

Long Short-Term Memory (LSTM) cells are a type of recurrent unit that can be employed in RNN. The study behind this type of cells tried to solve the problem of vanishing gradient that can occur for a traditional RNN.
As the name says, these cells have a sort of "memory" to model long-range dependencies. In Figure 2.16, a LSTM cell is depicted.

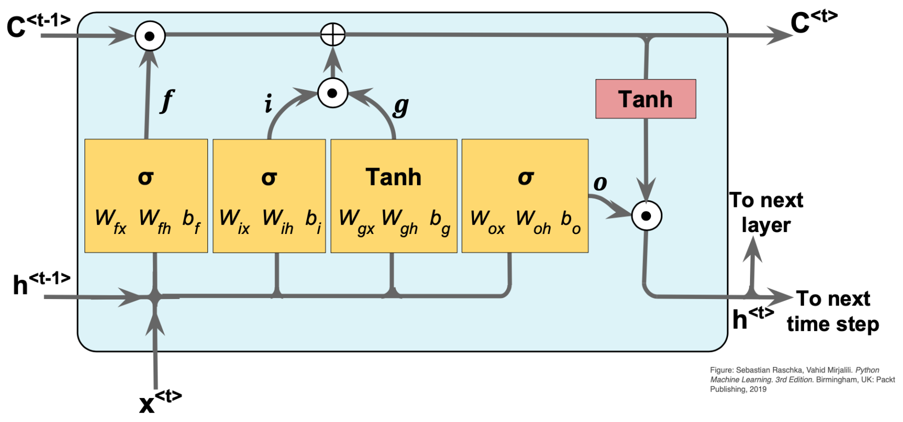


Figure 2.16: LSTM cell [28]

On top of the links to the other neurons, a cell is composed of an input node i, an input gate g, a forget gate f and an output gate o. Moreover activation functions are employed such as sigmoid (σ) and hyperbolic tangent (tanh).
The cell is defined by a state, denoted as C, which is updated according to the indications arriving from input and forget gates. The updated state then contributes to the new value in the hidden unit through the output gate.
Each gate acts like an activation function. First, the input gate controls if a new value enters the cell. The forget gate chooses if the state is "memorized" or needs to be "forgotten". Finally, the output gate decides if the state can be employed to compute the output activation of the LSTM cell. Each connection of the cell and of the gates are associated weights and bias like for the Fully-Connected hidden layer neurons, which need to be learned during the training phase.

Training of the network

The goal is to find the optimal values for $\mathbf{W}$ and $\mathbf{b}$ in order to:

- map in the best way the input-output relationship, if a regression problem is considered
- providing the correct label to the input considered, if the problem is related to a

classification

Thus a minimization problem is considered, where the total loss function is the average of every losses computed for every sample i. ARIEL focuses on a classification, therefore, considering N samples, the output $\mathbf{y_i}$ has to match the label prediction done $\mathbf{a_i} = f(\mathbf{W}\mathbf{x_i} + \mathbf{b})$ and the loss function can be written as Equation 2.5.

$$J = \frac{1}{N}\sum_i J_i\left(\mathbf{y_i}, f\left(\mathbf{W}\mathbf{x_i} + \mathbf{b}\right)\right) \tag{2.5}$$

Many different formulations can be chosen. In this dissertation, the functions employed are part of the Cross Entropy functions which are particularly indicated when the output is a prediction probability. This type of loss in particular increases as the prediction is further from the actual label. Therefore it tries to converge where these prediction match the labels imposed.
This loss function is obtained relating the output with the probability associated:

$$J = -\sum_{c=1}^{M} y_{c,o}\ln(P_{c,o}) \tag{2.6}$$

where M classes are considered, y is a binary indicator if class c is the correct classification for sample o and P predicted probability that sample o is of class c.

Moreover an additional metric on top of the loss can be defined. Having another metric allows to have another indication on the performance of the network. The most employed is the Accuracy: this indicator shows how often the prediction is correctly classified. It is in fact evaluated as the ratio between the number of correct predictions and the total number of predictions. In the case of a binary classification, e.g. Positive-Negative, it becomes:

$$Acc = \frac{T_P + T_N}{T_P + T_N + F_P + F_N}$$

where T_P and T_N represent, respectively, the true-positive and true-negative predictions while F_P and F_N the false-positive and false-negative predictions, respectively. This formula can be easily extended also to a multi-class problem.
This metric can also be displayed in the form of a Confusion Matrix: this matrix relates the actual labels with respect to the predictions done. The diagonal elements display the corrected predictions, while false positives or false negatives are found in the off-diagonal terms.

Before the minimization process can start, it is necessary to select a proper initialization for the variables of interest, **W** and **b**, which can be done in different ways. One approach would be to set them equal to small random numbers with Gaussian distribution, however, during training, the gradients may be flattened and thus preventing the improvement of the model. The most used initialization involves the scaling of the variables on the square root of the number of the inputs, the so-called "Xavier initialisation". It is evaluated as a uniform probability distribution between the range $[-\frac{1}{\sqrt{n}}; \frac{1}{\sqrt{n}}]$, with n the number of inputs to the considered neuron:

$$w_0 = U\left([-\frac{1}{\sqrt{n}}; \frac{1}{\sqrt{n}}]\right)$$

Another option is to normalize the inputs with respect to their considered batch. This procedure is called "batch-normalization" and allows to improve speed and performances of the training phase. After the input is normalised with respect to expected value and variance of the selected batch, it is scaled and shifted so as to grant flexibility to the training.

During the minimization, the parameters **W** and **b** are corrected using the Gradient Descent Algorithm: by computing the gradients of the loss function and combining it with a hyperparameter - the learning rate α - it is possible to update **W** and **b**, as elucidated in Equation 2.7.

$$\begin{cases} \mathbf{W_{i+1}} = \mathbf{W_i} - \alpha \dfrac{\partial J}{\partial \mathbf{W}^T} \\ \mathbf{b_{i+1}} = \mathbf{b_i} - \alpha \dfrac{\partial J}{\partial \mathbf{b}^T} \end{cases} \tag{2.7}$$

It is to be noted that the hyperparameter α controls the step size of the Gradient Descent. The value has to be chosen a priori and tuned accordine to the algorithm performance, considering a trade-off between convergence speed and chance of minima over-shooting. It is also possible to employ a variable learning rate α: it can be set to decay after a certain number of training epochs in order to converge as close as possible to the minima. The decay can be for example exponential or staircase-like.
In addition, in more complex scenarios, the loss function can be non-convex, thus convergence to a local minima may be more difficult. Hence some revised minimization procedures can be considered:

Batch Gradient Descent : The overall training set is considered at each epoch.

Mini-batch Gradient Descent : The training set is divided in batches which, at each step, are analysed iteratively. It is recommended if the dataset is significantly large even though this process is more noisy than Batch Gradient Descent.

Stochastic Gradient Descent (SGD) : At each epoch, only one training batch, selected randomly, is considered in the loss function. Even if this process is time-consuming, it is often preferred.

SGD is the most popular optimizer, however due to its many drawbacks, many variants have been studied with the aim of improving the algorithm efficiency. Possible modifications to SGD are the following:

SGD with momentum : An additional term is inserted in the computation of the gradient, the momentum. This allows to avoid the SGD to get stuck in local minima or saddle points and to speed up the algorithm.

Adagrad : Instead of a simple momentum term, an estimate of the squared gradient is used to reduce the gradient. Therefore, the steps size gradually decreases until the minimum is reached. However, if saddle points are present, this algorithm may get trapped inside.

Root Mean Square Propagation (RMS-Prop): The gradient is computed using a exponential moving average, expressed through a decay rate.

Adam : This algorithm uses the contributions in Adagrad but evaluated through a exponential moving average, like RMS-Prop does. In this case however, two decay rates are considered, one for the gradient and one for the squared gradient.

The Adam method is one of the most efficient optimizer, because it combines the advantages of Adagrad's method with those of RMS-Prop. For this reasons, it is also the most extensively used method in the literature.

Eventually, the gradients used to compute the updated values for $\mathbf{W}$ and $\mathbf{b}$ are built taking into account all the neurons in between input and output layer. This process is called back-propagation: starting from the output layer, gradients are computed considering the derivative of the loss with respect to the activation output and then exploiting the chain rule. Figure 2.17 shows this procedure with respect to the forward propagation, corresponding to the above-mentioned steps in a neuron. In this case, x and y may correspond to weights and bias and L to the loss function.

This sequence of operations is part of the so-called "training" of the NN and is an essential step for the building of the network.
The training requires a complete dataset, composed of samples and corresponding labels (if considering a classification problem).
The overall dataset is divided in subsets such as Training, Testing and Validation sets. The first subset is used during training to properly build the model needed. The second

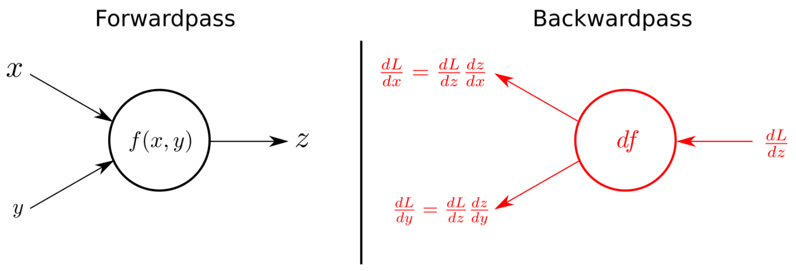

Forwardpass
Backwardpass
x
y
$f(x,y)$
z
df
$\frac{dL}{dx} = \frac{dL}{dz}\frac{dz}{dx}$
$\frac{dL}{dy} = \frac{dL}{dz}\frac{dz}{dy}$
$\frac{dL}{dz}$

Figure 2.17: Forward and backward propagation [29]

allows to verify the capabilities of the model afterwards, by feeding data that was not analysed during training. The Validation subset is an additional one to keep track of the performances of the model during training: this can be helpful to avoid the so-called "overfitting".
The typical proportion of dataset used are 70% for Training and 30% for Testing, but if Validation is also considered, the ratios become 60%-20%-20%.

Bias-Variance trade-off

To ensure that the model designed works correctly, many careful considerations have to be taken into account, first and foremost the so-called bias-variance trade-off [30].
Bias and variance are important characteristics in machine learning and greatly influence the success in training the network. These features are linked to how the Training and Validation losses behave during training. These variations can indicate two possible scenarios:

- If the training loss is much lower than the validation, the model is **overfitting**. In fact, high variance means that the model tries to perfectly classify each samples.
- If the training error is higher than expected, the model is far too general and thus misses most of the link between samples and labels, hence it is **underfitting**. This is due to the high bias in the model.

The aim is to have a model with both low bias and low variance. Figure 2.18 perfectly shows this phenomenon.
To obtain low bias different strategies can be used such as:

- Enlarging the model, by adding more hidden layers or more neurons for each layer so that the loss function has more degrees of freedom. Doing so, the model can manage non-linearities in a better way
- Training longer

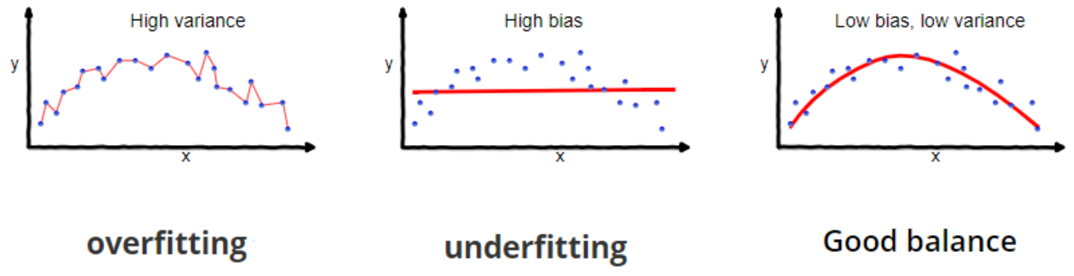


Figure 2.18: Bias-variance tradeoff [31]

- Searching for another network architecture that can better suit the problem

Instead, the variance can be reduced by:

- Searching for a more appropriate network architecture, just like for the bias
- Using a larger dataset as input. If not possible, data augmentation can be a possible substitute
- Using regularization

Regularization aims at penalizing the loss function by adding a term linked to a regularization parameter λ as shown in Equation 2.8.

$$J = \frac{1}{N}\sum_i J_i\left(\mathbf{y_i}, f\left(\mathbf{W}\mathbf{x_i} + \mathbf{b}\right)\right) + \frac{\lambda}{2N}R(\mathbf{W}) \tag{2.8}$$

In this way, if λ is large, higher weights are penalised, in order to increase the contribution of the neurons to the final loss. Different formulations of $R(\mathbf{W})$ can be used however, the most popular are L_2 and L_1 regularizations, defined in Equation 2.9.

$$\begin{aligned} L_2 \text{ regularization: } R(\mathbf{W}) &= \sum_k\sum_l \mathbf{W}_{k,l}^2 \\ L_1 \text{ regularization: } R(\mathbf{W}) &= \sum_k\sum_l |\mathbf{W}_{k,l}| \end{aligned} \tag{2.9}$$

Another form of regularization is the so-called dropout: as the name states, it is set through a probability of eliminating a neuron from the network, leading to a smaller network. It is to be noted that the eliminated neuron comes back at the next training step. Therefore its weight is spread out across the other inputs, decreasing the influence of the dropped neuron, just like the term $\frac{\lambda}{2N}R(\mathbf{W})$ does in the loss function.

Introduction to the Siamese Model

So far, only Sequential models have been described meaning that layers are assembled one after the other from the input to the output layer. However different model structures can be exploited.
In this dissertation, a Siamese Network is put in place in SIERRA. By definition, this model compares inputs with respect to a reference, which makes it perfectly suited for the problem at hand.
The reference model is the one discussed by H. Essam and S.L. Valderrama for Keras [32], which takes as reference the well-known FaceNet network [33].
The first step is the re-arrangement of the dataset. In fact, the overall samples are first divided in three subsets, named "Anchor", "Positive" and "Negative": the former is associated to the reference the model performs inference on, while "Positive" and "Negative" refer, respectively, to the closest and farthest result of this inference.
Then, an embedding model is developed which allows to extract features from the inputs. In the model used in Keras, a pre-trained CNN is attached to Fully-Connected layers to properly distinguish the different classes. The outputs of the overall network are the distances Anchor-Positive and Anchor-Negative, and the goal is to minimise the former and maximise the latter. These distances are evaluated in a customised layer as follows:

$$\begin{aligned} ||AP|| &= ||Emb(A) - Emb(P)||^2 \\ ||AN|| &= ||Emb(A) - Emb(N)||^2 \end{aligned} \tag{2.10}$$

where A refers to Anchor, P to Positive, N to Negative and $Emb(\cdot)$ to the embedding applied.
In this case, the loss function is the Triplet Loss, depicted in Figure 2.19 and described by the cost function in Equation 2.11.

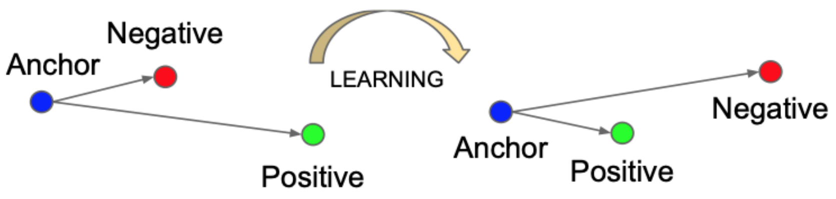


Figure 2.19: Triplet loss function working principle [33]

$$J = \max\left(||AP|| - ||AN|| + m, 0\right) \tag{2.11}$$

where m is a margin to be enforced between the distances computed.

As a final step, a different metric can be used to define how close the Anchor is to the Positive or Negative, called Cosine Similarity. This metric is computed from the normalized dot product of the labels **l** with the predictions **p** made:

$$CosineSimilarity = \frac{\mathbf{l} \cdot \mathbf{p}}{||\mathbf{l}||\,||\mathbf{p}||}$$

If the embeddings are made correctly the Cosine Similarity in the Anchor-Positive case will be much larger than for the Anchor-Negative.

3 | Data pre-processing

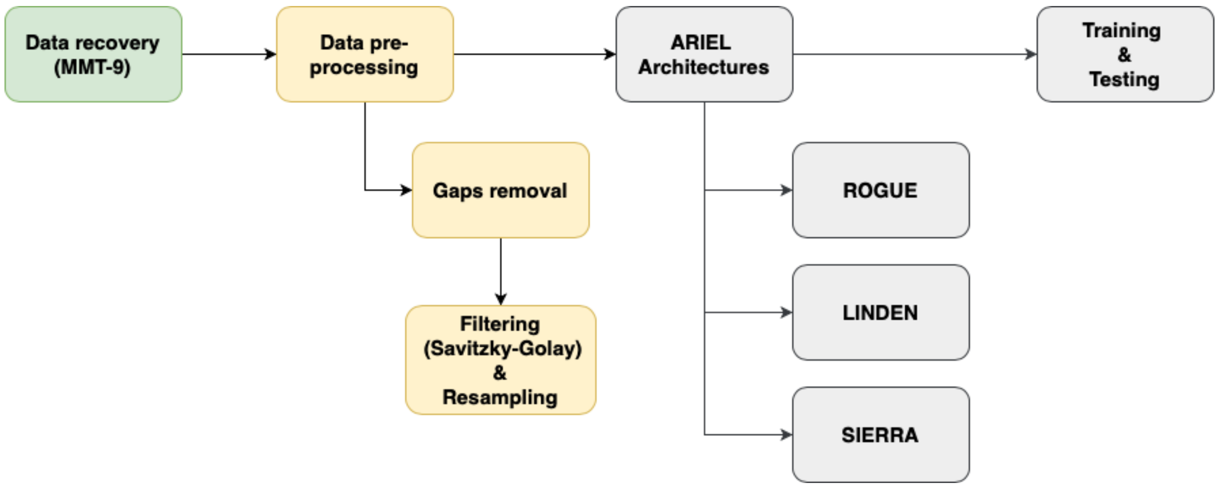


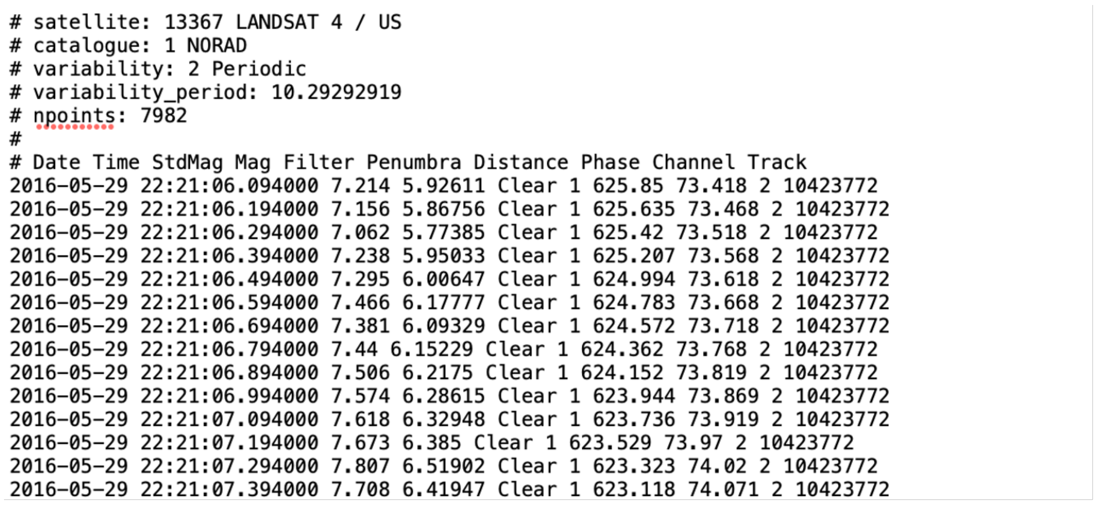
Figure 3.1: Thesis workflow - Light curves recovery and pre-processing

This chapter, in fact, gives an overview of the light curves' pre-processing phase obtained from the MMT-9 database (Figure 3.1). First of all, the data recovery, with a focus on the orbital regime and the tumbling motion, is described. The steps of pre-processing are reported afterwards; in particular, the removal of possible gaps, the Savitzky-Golay (SavGol) filtering and resampling. Eventually, light curves are ready for ARIEL.

3.1. Data recovery

The MMT-9 observatory, described in Section 2.1, provides light curves from non-military space objects launched by countries other than the Russian Federation [20]. Figure 3.2 gives an example of a input from the database and Table 3.1 lists the parameters depicted.

The recovered data considered satellites which light curves have been observed up to the 22nd of April 2022. Over 6.314 space objects complemented by over 150.000 light curves have been collected and catalogued in two different Python JSON files showcased in Table 3.2 with the most relevant features. Many considerations have to be done regarding this



```
# satellite: 13367 LANDSAT 4 / US
# catalogue: 1 NORAD
# variability: 2 Periodic
# variability_period: 10.29292919
# npoints: 7982
#
# Date Time StdMag Mag Filter Penumbra Distance Phase Channel Track
2016-05-29 22:21:06.094000 7.214 5.92611 Clear 1 625.85 73.418 2 10423772
2016-05-29 22:21:06.194000 7.156 5.86756 Clear 1 625.635 73.468 2 10423772
2016-05-29 22:21:06.294000 7.062 5.77385 Clear 1 625.42 73.518 2 10423772
2016-05-29 22:21:06.394000 7.238 5.95033 Clear 1 625.207 73.568 2 10423772
2016-05-29 22:21:06.494000 7.295 6.00647 Clear 1 624.994 73.618 2 10423772
2016-05-29 22:21:06.594000 7.466 6.17777 Clear 1 624.783 73.668 2 10423772
2016-05-29 22:21:06.694000 7.381 6.09329 Clear 1 624.572 73.718 2 10423772
2016-05-29 22:21:06.794000 7.44 6.15229 Clear 1 624.362 73.768 2 10423772
2016-05-29 22:21:06.894000 7.506 6.2175 Clear 1 624.152 73.819 2 10423772
2016-05-29 22:21:06.994000 7.574 6.28615 Clear 1 623.944 73.869 2 10423772
2016-05-29 22:21:07.094000 7.618 6.32948 Clear 1 623.736 73.919 2 10423772
2016-05-29 22:21:07.194000 7.673 6.385 Clear 1 623.529 73.97 2 10423772
2016-05-29 22:21:07.294000 7.807 6.51902 Clear 1 623.323 74.02 2 10423772
2016-05-29 22:21:07.394000 7.708 6.41947 Clear 1 623.118 74.071 2 10423772
```

Figure 3.2: Extract of MMT-9 entry data [20]

Entry parameter	**Description**
Satellite	Includes name, ID and the country of origin of the space object. If the object is a debris or a rocket body it is specified with the flag "DEB" or "R/B" respectively
Catalogue	Public catalogue where the orbital parameters are obtained: NORAD [34] or McCants [35]
Variability, Variability period	If the satellite displays a periodicity in its attitude state: it can be periodic, aperiodic and non-variable. For the former case, the period is specified, while for the others it is set to 0
Npoints	Number of records acquired for that particular object
Date, Time	Date and time of the light curve acquisition
StdMag	Standard magnitude: magnitude normalised at 1000 km distance and 90° phase angle
Mag	Apparent magnitude
Filter	If a filter is applied during the acquisition
Penumbra	If the satellite is in penumbra or not
Distance, Phase	Changing distance and phase angle with respect to the observatory during the space object passage
Channel	Which channel is involved during the acquisition
Track	Identification number for the light curve acquired

Table 3.1: Parameters shown in a typical file from the MMT-9 database [20]

Satellite characteristics
Name of the space object
Associated NORAD Id
Orbital regime
Type of satellite: Debris, Rocket body, Satellite
Attitude regime: Periodic, Aperiodic, Non variable
If Periodic, the Tumbling period is specified
Track ids
Number of tracks

(a) Satellite information

Light curves
Track ids
Acquisition data: Date and Time of acquisition Apparent magnitude

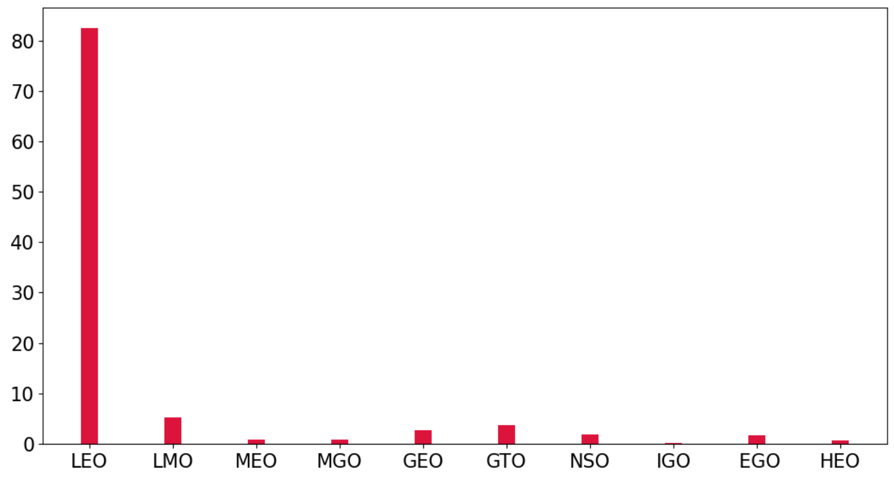
(b) Light curves information

Table 3.2: JSON files with the most significant data collected

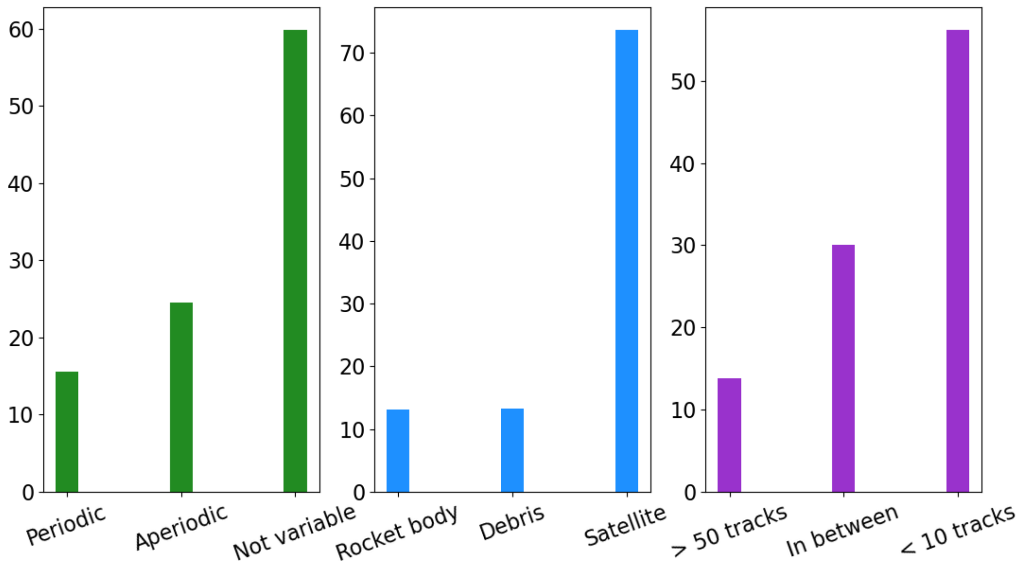
selection of parameters. To start with, the orbital regime has been obtained thanks to the NORAD ID provided in the files. Through catalogue matching with Celestrack [36] and McCants [35] databases, the orbital elements are recovered. Thus, the object's orbit is extrapolated from those parameters, according to the definitions provided by ESA and summarized in Appendix A.1. Worth noticing that the entry "Track ids" allows to trace back the collected files.
In Figure 3.3, it is represented the distribution of data in the files regarding orbital and attitude regime, the type of object and number of light curves available per object.
Figure 3.4 shows how dataset sorting affects the variability of space objects in a specified orbital regime. Precise values can be found in Appendix A.2.
From this preliminary analysis, it appears that LEO objects, mainly Satellites, populate the catalogue. Something to be wary is that many objects have less than 10 tracks available, which grant some kind of variability to the dataset but it might be limiting from a characterisation stands-point. In fact, too much variance in the dataset lead NN underfitting as discussed in Section 2.2.
Regarding the attitude regime, non-variable objects represent a substantial part of the dataset. However, they become of little interest due to the low number of light curves available and high bias towards satellites.

In this work, just a fraction of the dataset is considered, focusing on objects:

- in the LEO or LMO region



(a) Orbital regime, (from definitions in Appendix A.1 [2])

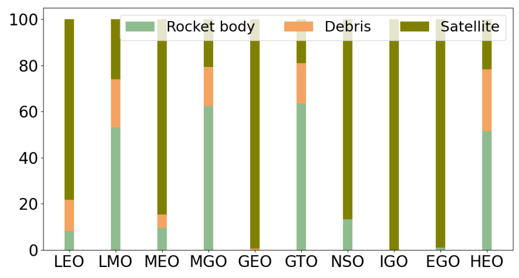


(b) Type, attitude regime and number of tracks

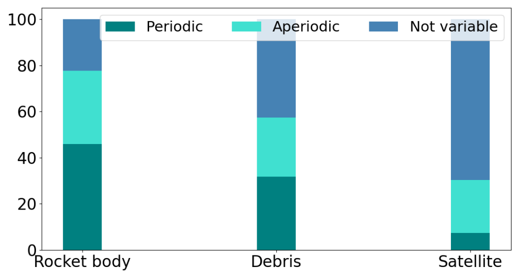
Figure 3.3: File criteria distribution with respect to the total number of space objects [%]

- having a periodic or aperiodic variability
- having sufficient light curves to create complete datasets

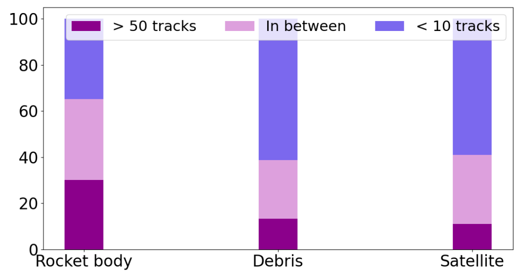
The selection fell on these features as they were deemed of interest for this dissertation. Therefore, only the data that matched these criterion were considered from here on.



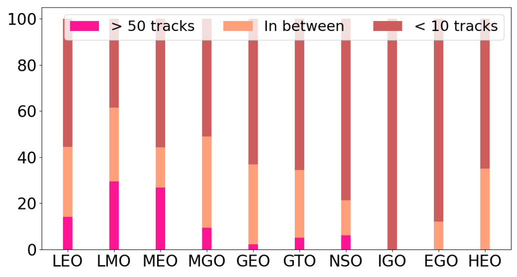
(a) Type of object vs Orbit



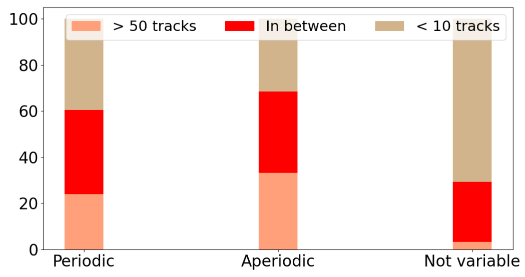
(b) Type of object vs Attitude regime



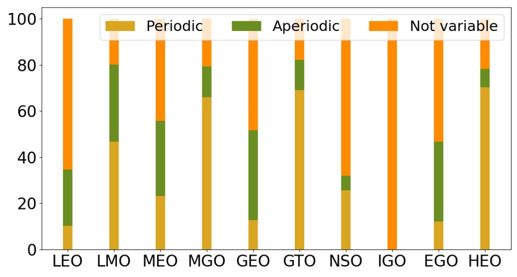
(c) Number of tracks vs Type of object



(d) Number of tracks vs Orbit



(e) Number of tracks vs Attitude regime



(f) Attitude regime vs Orbit

Figure 3.4: Data Distribution [%]

3.2. Pre-processing

Light curves directly obtained from the MMT-9 database cannot be directly employed in ARIEL being raw data. Hereafter, the steps to make the dataset comply with the NN are listed:

1. Remove data gaps in light curves
2. Filter light curves using the SavGol filter, elucidated in Section 2.1
3. Resample the signal according to the MMT-9 documentation [20]

Gaps removal

In some cases, light curves present gaps to be removed before pre-processing. In Figure 3.5, for instance, the original light curve is divided in different light curves. As a result, the initial number of light curves tends to increase after this operation.

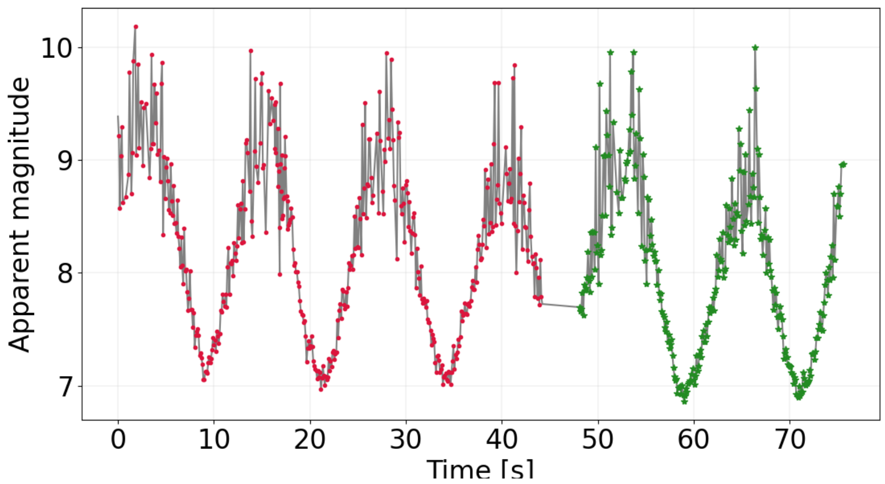


Figure 3.5: Example of removal of gaps in light curve: two light curves are recovered - the red and the green one - from the original in grey

Filtering and resampling

Light curves are not noise-free; hence the Savitzky-Golay (SavGol) filter cleans up the input signal. Extensively described in Section 2.1 and adopted in Python with the function `savgol_filter`, this smoothing filter depends mainly on two parameters: the window length w and order o.

Provided that $w \geq (o+1)$, different configurations for w and o are considered:

$$w = \left[\frac{\Delta t}{4}; \frac{\Delta t}{3}; \frac{\Delta t}{2}\right] \text{ with } o = 5$$
$$o = [2; 3; 4; 5] \text{ with } w = \frac{\Delta t}{2}$$

with Δt being the light curve duration. Figure 3.6 gives an example of the filtering procedure in the aforementioned configurations. For the sake of clarity only a portion of the signal is depicted.
It can be noted that the larger the window, the more the signal is smoothed. The same occurs considering a smaller polynomial order, even if the difference is less pronounced. The final configuration selected therefore considers:

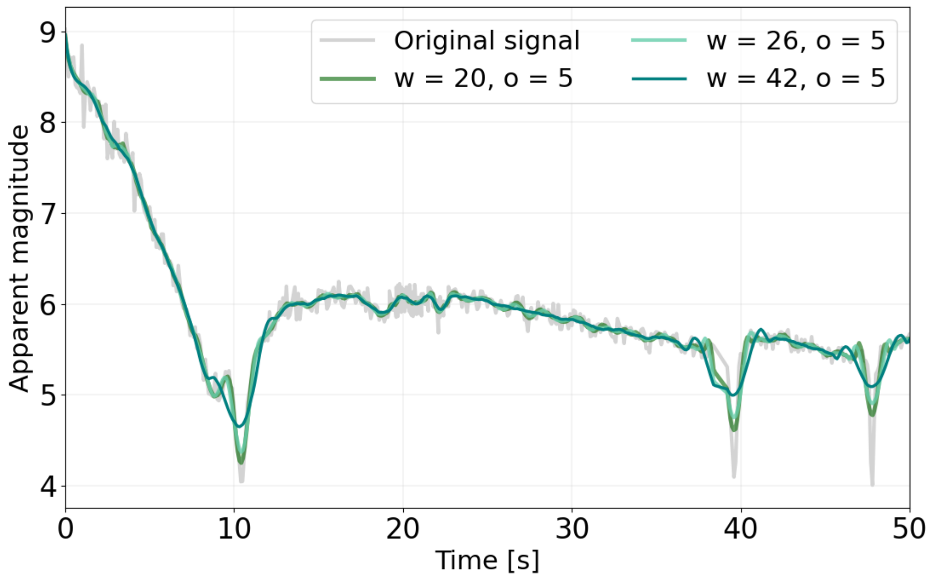
$$w = \frac{\Delta t}{2} \qquad\qquad o = 3$$

This parameter selection in fact grants the desired smoothing effect without fitting noise at the extremities. Figure 3.7 depicts the previous example with the selected pair (w, o). While filtering, resampling at 0.1 s is performed, as indicated in the MMT-9 documentation [20]. This can be easily done using the function `numpy.interp`.

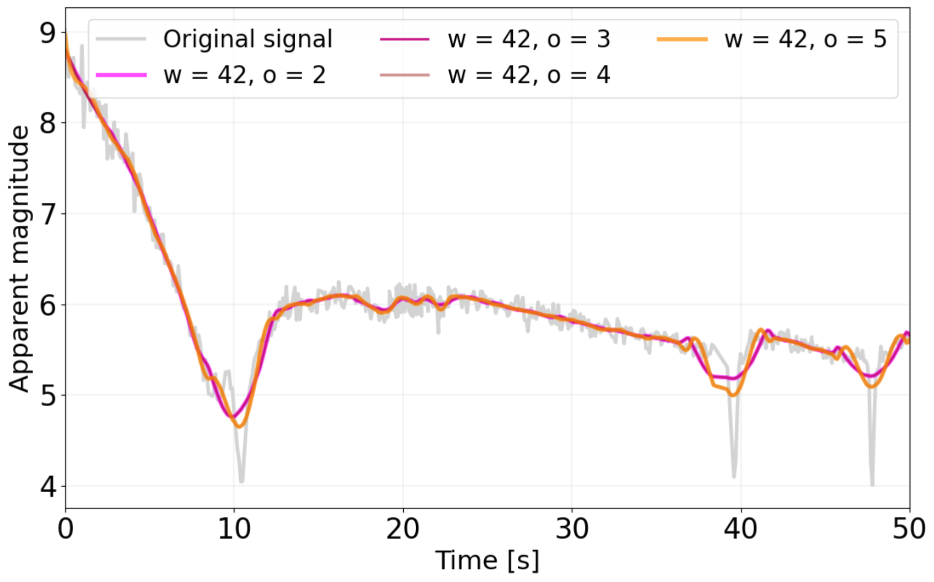
After all these steps, the datasets for ARIEL can be prepared. The features that are kept are the following:

- Name of the space object
- Type of object considered: Rocket body, Debris, Satellite
- Duration of the acquisition of the light curve considered
- Apparent magnitude

To be noted that after those routines, only light curves lasting at least 20 s are actually kept, in order to perform a meaningful analysis.



(a) Fixed o, variable w



(b) Fixed w, variable o

Figure 3.6: SavGol filter application on a light curve

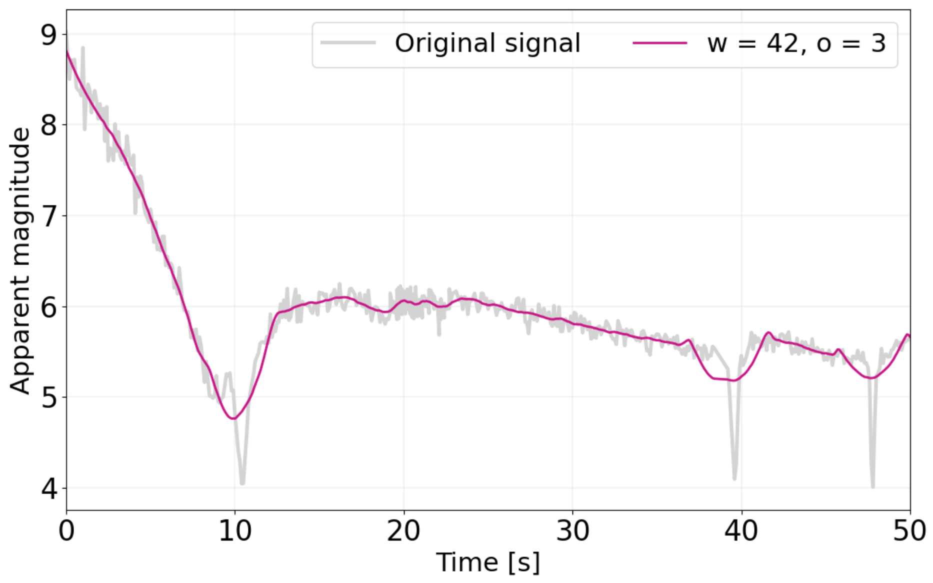


Figure 3.7: Filtering procedure in action employing the selected (w, o)

4 | Proposed architectures and NN setup

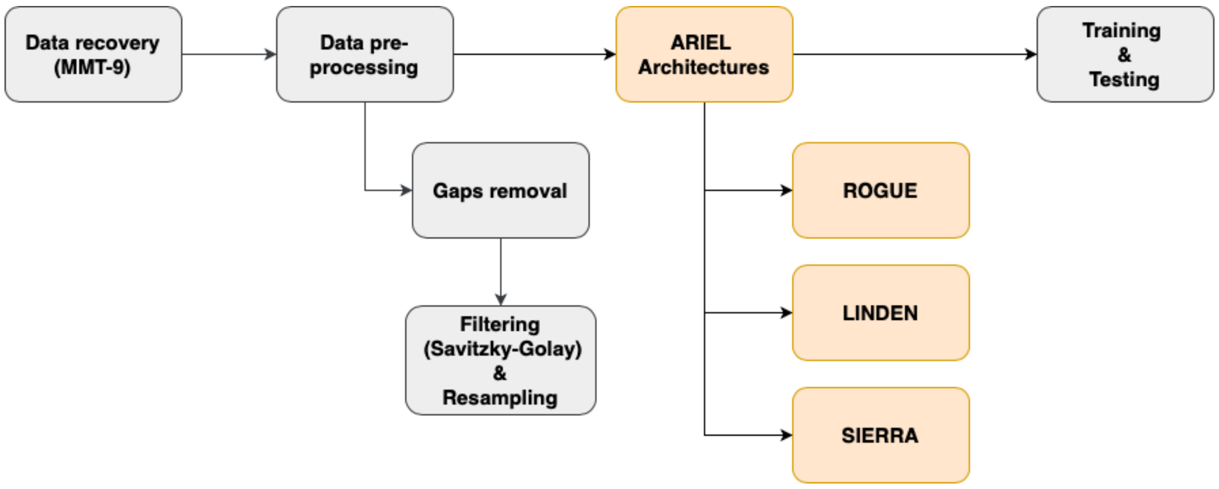


Figure 4.1: Thesis workflow - Architectures

This chapter embodies the core of this dissertation as shown in Figure 4.1. The different strategies employed, which concern feature extraction and correlation evaluation, are in fact described.
The first NN, denoted ROGUE, is devoted to telling a Rocket body apart from Satellites and Debris.
This NN laid the foundations for the Feature extraction segment which is improved and combined with a final correlation layer in LINDEN and a Siamese network in SIERRA. All the strategies have been implemented using Python library Tensorflow in particular Keras.

4.1. ROGUE: One type recognition

The **R**ocket b**o**dies Li**g**ht c**u**rves Id**e**ntification (ROGUE) NN aims at recognizing Rocket bodies among a different pool of space objects. It was built to assess the capabilities of a mixed CNN - LSTM network to classify a particular type of object. In particular Rocket bodies are firstly covered featuring an easier to identify geometry. The classification problem follows the hereafter rule.

$$\text{Labels} = \begin{cases} 1 \text{ if the light curve belongs to a Rocket body} \\ 0 \text{ otherwise} \end{cases}$$

As discussed in Section 1.2, CNNs have been extensively used for the analysis of light curves. Similarly, RNNs - in particular LSTMs - have proven to be quite proficient in classifying sequence data.
This NN implementation allows extracting the input signal features, as well as the data time dependence. The overall model employed is described in Table 4.1.

Layer	**Features**	**Number of parameters**

Embedding	Input dimension = 1.000;
	Output line = 20;
	Activated masking
		20.000

1D Convolutional	Filters = 64;
	Kernel size = 3;
	Padding = same
		3.904

Batch Normalisation		256
ReLU activation		

1D Convolutional	Filters = 64;
	Kernel size = 3;
	Padding = same
		12.352

Batch Normalisation		256
ReLU activation		
LSTM cell	Units = 64; Return sequences = True	33.024
Batch Normalisation		256
LSTM cell	Units = 32; Return sequences = True	12.416
Global Max Pooling (1D)		

Fully-Connected	Units = 10;
	Activation = ReLU;
	L2 regularization applied with $\lambda = 0.001$
		330

Fully-Connected	Unit = 1	11

Table 4.1: ROGUE NN breakdown structure

Instead of a simple input layer, an Embedding layer processes the input light curve, while also removing all the padding done while preparing the dataset (cf. Section 4.4).
The light curve is then analysed by a set of 1D Convolutional layers, blended with Batch Normalisation and ReLU activation layers, retaining the general features from light curves. Then a set of LSTM cells analyses the peculiar time-dependent traits of the data. The "Return sequences" option is activated to stack more LSTM layers.
The 1D Global Max Pooling layer is then followed by the last two Fully-Connected layers. The working principle is analogous to the Max Pooling layer, but the pooling operation extends to the overall time dimension.
Additionally, the first Fully-Connected layer applies a L_2 regularization to reduce the risk of over-fitting. The output is directly available from the last Fully-Connected layer. Overall, over 82.000 parameters are properly tuned along the training phase.

4.2. LINDEN: Feature extraction and Correlation evaluation

The **L**ight curve **I**de**n**tification and Correlation (LINDEN) architecture is devised to classify a space object, and possibly correlate it to a catalogued one. Assuming that an a-priori prediction on the object observed is available, its light curve can be retrieved thanks to the database outlined in Section 3.1. Thus, a correlation degree between the light curves of the unknown object and of the predicted object can be evaluated. Figure 4.2 depicts the steps above-mentioned.

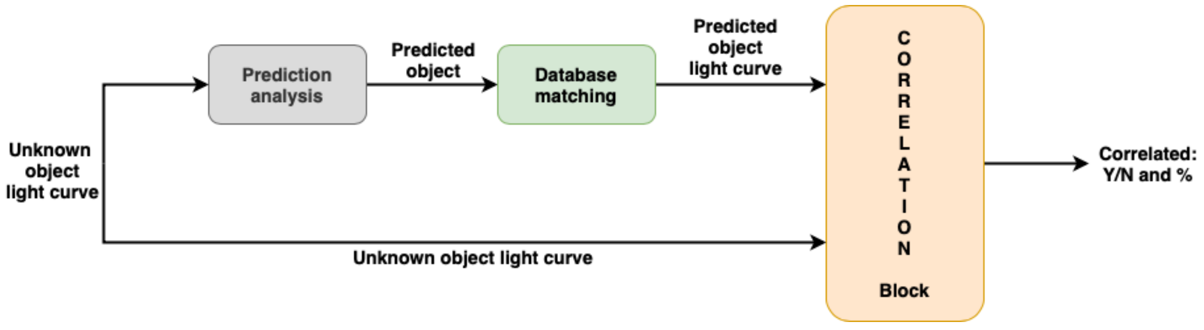


Figure 4.2: Block scheme of the baseline for LINDEN

The core of this strategy is the Correlation block: it extracts features from both light curves so as to give a prediction of the type of object; afterwards, it establishes the correlation degree between the two.
This block is detailed in Figure 4.3.
As previously anticipated, LINDEN is divided in two parts: the Feature extraction block

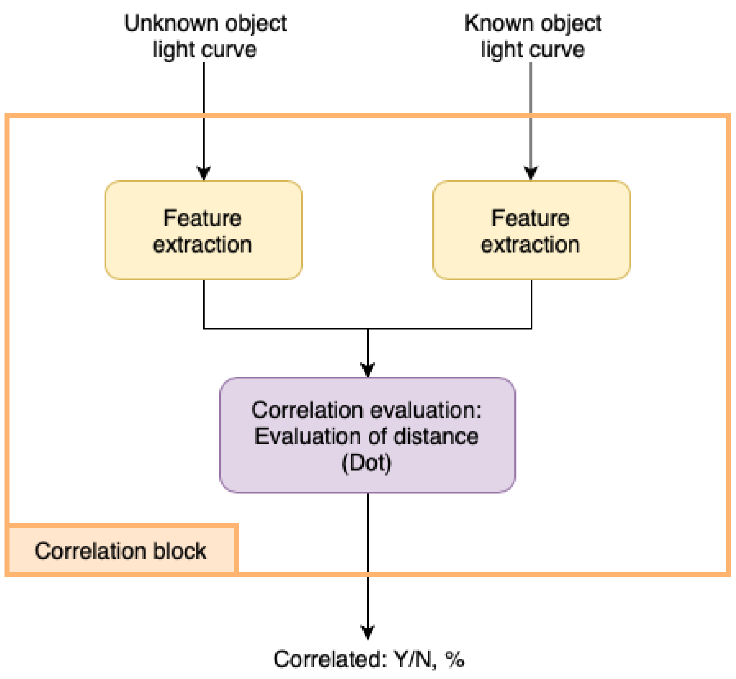


Figure 4.3: Correlation block of LINDEN

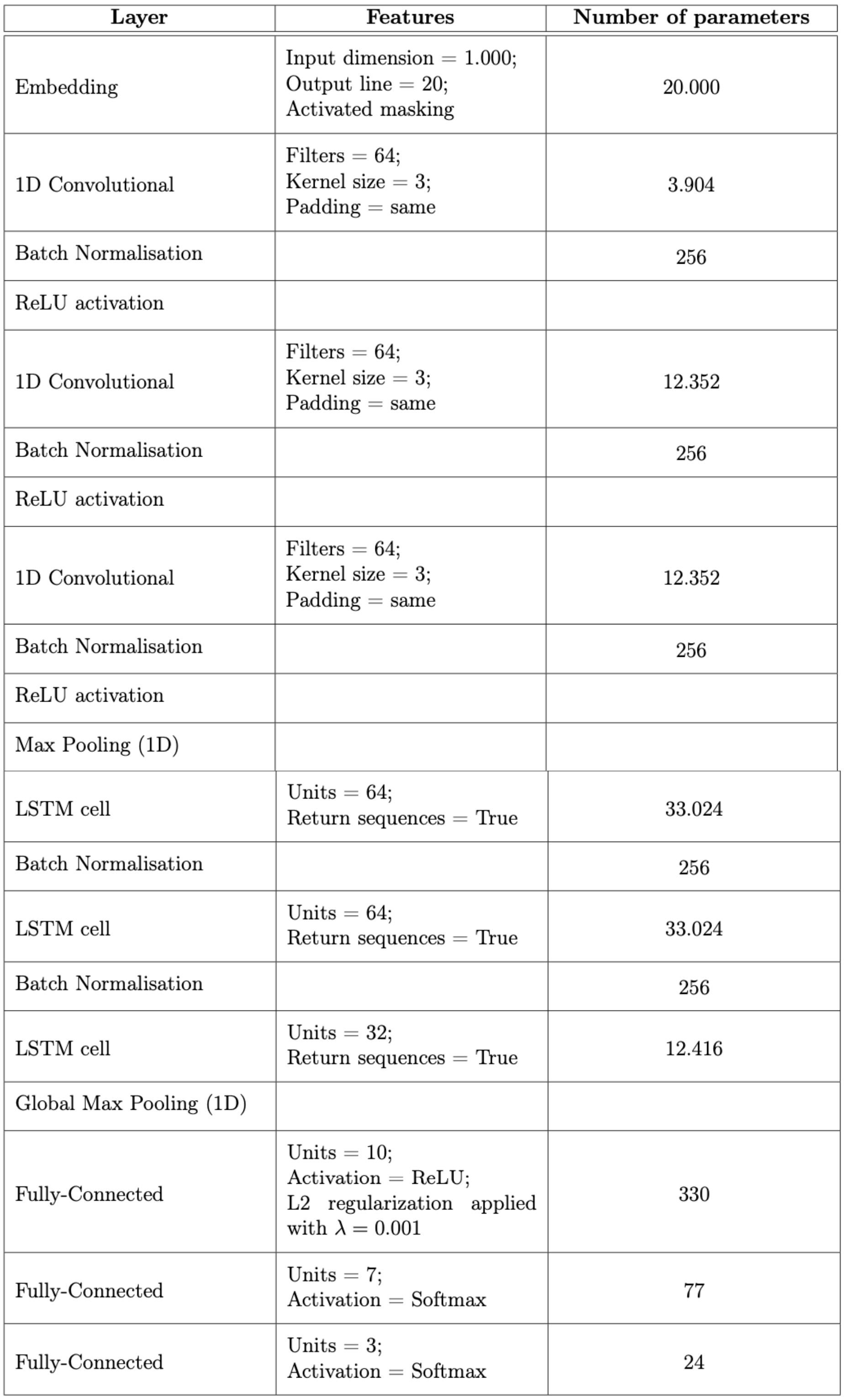
and the Correlation evaluation block. The Feature extraction block is responsible for processing the input light curve, while the Correlation evaluation compares the outputs.

Feature extraction

In Figure 4.4 are reported the layers employed for the Feature extraction part.
The Feature extraction blocks have the aim to process the input light curve and assign the following labels:

$$\text{Labels} = \begin{cases} 0 \text{ if the light curve belongs to a Debris} \\ 1 \text{ if the light curve belongs to a Rocket Body} \\ 2 \text{ if the light curve belongs to a Satellite} \end{cases}$$

LINDEN Feature extraction block is nothing but an augmented version of ROGUE. In fact, another Convolutional layer, always accompanied by Batch Normalisation and ReLU activation, adds to a 64-unit LSTM cell.
Being a multi-class classification, the network necessitates over 128.000 trainable parameters.
In the first Fully-Connected layer, the L_2 regularization is still applied. The following layers have a Softmax probability distribution over the three labels. For example, a light curve can be associated at 70% to a Rocket body, 20% to a Debris and 10% to a Satellite.



Layer	Features	Number of parameters

Embedding	Input dimension = 1.000;
	Output line = 20;
	Activated masking
		20.000

1D Convolutional	Filters = 64;
	Kernel size = 3;
	Padding = same
		3.904

Batch Normalisation		256
ReLU activation		

1D Convolutional	Filters = 64;
	Kernel size = 3;
	Padding = same
		12.352

Batch Normalisation		256
ReLU activation		

1D Convolutional	Filters = 64;
	Kernel size = 3;
	Padding = same
		12.352

Batch Normalisation		256
ReLU activation		
Max Pooling (1D)		
LSTM cell	Units = 64; Return sequences = True	33.024
Batch Normalisation		256
LSTM cell	Units = 64; Return sequences = True	33.024
Batch Normalisation		256
LSTM cell	Units = 32; Return sequences = True	12.416
Global Max Pooling (1D)		

Fully-Connected	Units = 10;
	Activation = ReLU;
	L2 regularization applied with $\lambda = 0.001$
		330

Fully-Connected	Units = 7;
	Activation = Softmax
		77

Fully-Connected	Units = 3;
	Activation = Softmax
		24

Figure 4.4: NN breakdown structure for LINDEN - Feature extraction

Correlation block

After the training of the Feature extraction part, it is inserted in the overall LINDEN Correlation block.
This block is organised so as to receive two light curves, process each of them inside the Feature extraction block and finally perform the correlation between the inputs. As a result, the model has 2 inputs and 1 output. The assigned labels are:

$$\text{Labels} = \begin{cases} 1 \text{ if the light curve belongs to the same type of object} \\ 0 \text{ otherwise} \end{cases}$$

In Table 4.2, it is summarized the overall structure of LINDEN.

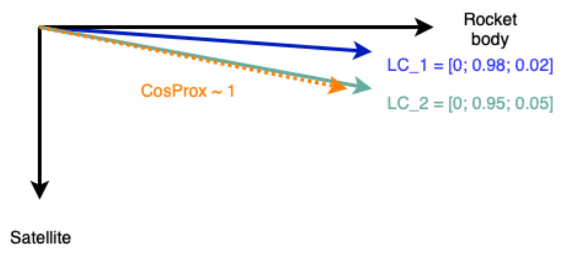
Layer	**Features**	**Number of parameters**
Input		
Feature extraction model	cf. Figure 4.4 Training = False	
Dot layer	Normalize = True	

Table 4.2: NN breakdown structure for LINDEN - Correlation evaluation

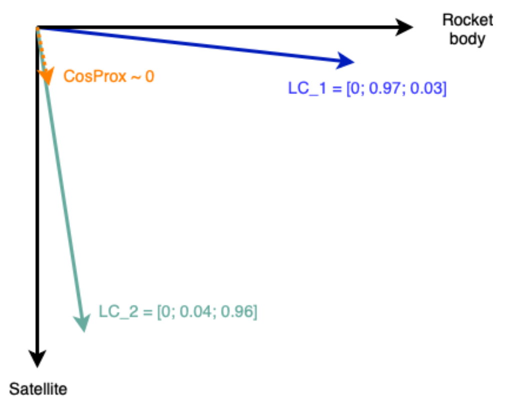
The Feature extraction block is not involved in the overall training of LINDEN. Thus, weights and biases adjusted during training are now frozen. Next, the correlation is evaluated through a dot product between the two output vectors from the Feature extraction model. With the "Normalize" option set to True, the result is the Cosine Proximity between the two vectors. Given two vectors $\mathbf{v_1}$ and $\mathbf{v_2}$, the Cosine Proximity is obtained as the cosine of the in-between angle, as shown in Equation 4.1.

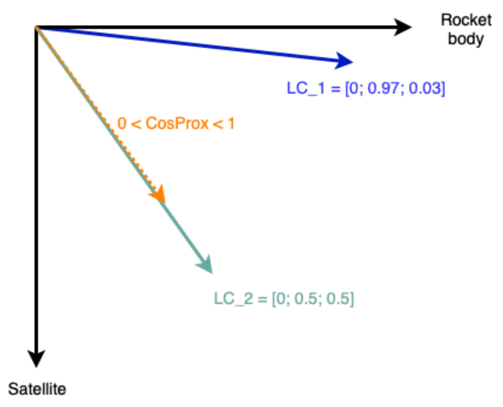
$$\mathbf{v_1} \cdot \mathbf{v_2} = ||\mathbf{v_1}||\,||\mathbf{v_2}||\cos(\angle\mathbf{v_1}\mathbf{v_2}) \Rightarrow \cos(\angle\mathbf{v_1}\mathbf{v_2}) = \frac{\mathbf{v_1} \cdot \mathbf{v_2}}{||\mathbf{v_1}||\,||\mathbf{v_2}||} \tag{4.1}$$

Each component of the output vector represents the probability that the light curves belongs to the corresponding type. Reprising the above-mentioned example, the prediction vector would become: $[0.2; 0.7; 0.1]$
respectively linked to the Debris, Rocket body and Satellite classes.
Figure 4.5 graphically shows the correlation degree with LC_1 and LC_2 being the prediction vectors from the baseline and the other considered objects.
Thus, while performing the Dot product, the resulting cosine will be closer to 1 or 0, as the output vectors are parallel or perpendicular respectively. In the first case, the light



(a) Same type





(b) Different types (c) Intermediate case

Figure 4.5: Cosine proximity visualisation (not to scale)

curves belong to the same category (Figure 4.5a) while in the second it is not the case (Figure 4.5b).
If the predictions are not close, like in Figure 4.5c, the cosine will have an intermediate value, with a coarse correlation.

As a consequence, the more accurate the type guess is, the easier it is for LINDEN to have a sharper correlation degree.
This factor has to be taken in consideration during the training phase of LINDEN Feature extraction block.

4.3. SIERRA: Siamese Network

Siamese N**e**twork Light curves Co**rr**el**a**tion (SIERRA) network is different from LINDEN but it still compares light curves and it determines if they belong to the same type. The Siamese Network, previously described in Section 2.2, directly performs feature analysis and distance evaluation all at once. This model is built according to the steps defined in Section 2.2. The embedding model employed is the Feature extraction block from LINDEN. The overall scheme of SIERRA can be shown in Figure 4.6 and summarized in Table 4.3.

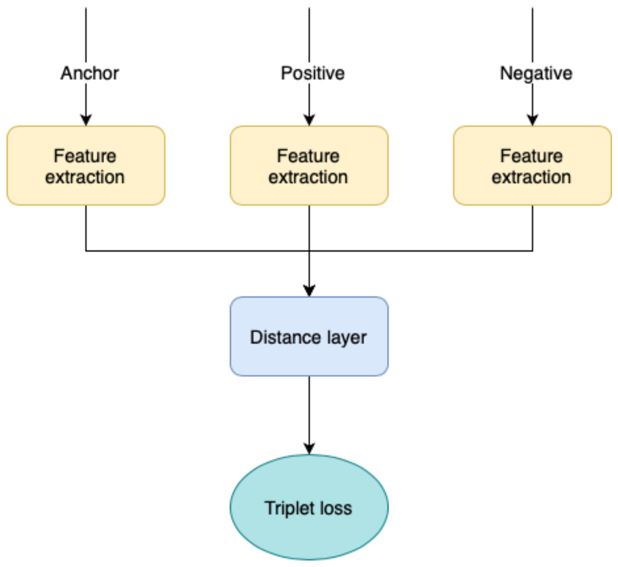


Figure 4.6: SIERRA NN structure

Layer	**Features**	**Number of parameters**
Input		
Feature extraction model	cf. Figure 4.4 Training = False	
Distance layer	Custom layer	

Table 4.3: SIERRA NN breakdown structure

As the model considered is a Siamese Model, the dataset is divided in three subsets:

$$\text{Labels} = \begin{cases} \textit{Anchor} \text{ the reference light curve} \\ \textit{Positive} \text{ if the light curve is of the same type as Anchor} \\ \textit{Negative} \text{ if the light curve is not of the same type as Anchor} \end{cases}$$

Three light curves are fed into the Feature extraction blocks to extract features and returns the prediction on the space object type.
Subsequently, the custom Distance layer computes how far apart the Anchor-Positive and Anchor-Negative are, according to Equation 2.10. These distances form the Triplet loss, defined in Equation 2.11. The aim is to get the Positive results closer while distancing the Negative ones.
Through the Cosine Similarity, Anchor and Positive should be close to 1 while between Anchor and Negative should be lower than 0.5.

4.4. Neural Network setup

Before training the above-mentioned NNs, it is important to consider some steps:

1. Prepare an adequate dataset
2. Choose the proper optimizer, loss function and metrics
3. Tune the hyper parameters accordingly, e.g. the learning rate

Dataset preparation

For each ARIEL network considered, datasets need to be prepared for the training and testing following the procedure detailed in Sections 3.1 and 3.2. Another aspect to take care of is ensuring a fixed input layer data length. Therefore, the light curves are zero-padded, which is the practice to add 0 to vectors so that they all have the same dimensions. Fortunately, those additional values are not considered in the NNs thanks to the Activated masking in the Embedding layer.
A problem that might also rise is that some datasets comprises a scarce number of light curves, hence limiting the training performances. Thus, the dataset is "multiplied", e.g. it is appended up to four times into the same file.
Eventually, the dataset is shuffled and divided in Training, Testing and Validation subsets with 60%-20%-20% ratios.

Different datasets have been devised: two regarding nominal conditions of operation of ARIEL and three others to assess the limits of these architectures, by considering many different platforms. All those datasets are denoted by "Nominal" or "Variability" respectively.

Nominal conditions

Two different datasets have been prepared and employed for the training and testing of each ARIEL NN:

Nominal_1 (Nom_1) : Includes only periodic space objects, one platform per type, with three objects in total. It contains 434 light curves, therefore it is multiplied 4 times.

Nominal_2 (Nom_2) : Encompasses both periodic and aperiodic objects, one platform per type and variability, with six objects overall. It includes 645 light curves, therefore it is multiplied 4 times.

The network is slightly biased towards the Rocket Body type as perceived in the results.

Variability test

Subsequently, the ARIEL network capabilities are further improved with different datasets characterised by a high platform-wise variability. In these datasets, both periodic and aperiodic spacecrafts are considered:

Variability_1 (Var_1) : 72 objects: 19 Rocket bodies, 37 Satellites and 16 Debris. It comprises 4334 light curves, therefore it is not repeated

Variability_2 (Var_2) : 40 objects: 9 Rocket bodies, 19 Satellites and 12 Debris. It has 1800 light curves, therefore it is multiplied 2 times.

Variability_3 (Var_3) : 20 objects: 6 Rocket bodies, 8 Satellites and 6 Debris. It contains 720 light curves, therefore it is multiplied 4 times.

As it can be noticed the number of platforms is not uniform among the three classes considered: this does not represent an issue as the number of light curves associated to each type is balanced out. In any case, if the unbalance compromises the training performances, class weights can be inserted as to have equal prediction probability for each label considered.

Optimization problem setup

The first step involves the choice in terms of optimizer, loss function and metrics which are more appropriate for the problem at hand. Table 4.4 summarizes these choices for the three networks described above.

Networks		Optimizer	Loss function	Metrics
ROGUE		Adam $\alpha = 10^{-4}$	Binary Cross Entropy	Accuracy
LINDEN	Feature extraction	Adam Variable α	Sparse Categorical Cross Entropy	Accuracy
	Correlation evaluation	Adam $\alpha = 10^{-4}$	Binary Cross Entropy	Accuracy
SIERRA		Adam $\alpha = 10^{-3}$	Triplet Loss	Cosine Proximity

Table 4.4: Optimization problem setup for ROGUE, LINDEN, SIERRA

Each choice was specifically tailored to the type of output the NN has.
The hyper parameter α also has to be selected accordingly. In general it has been reduced from the nominal value 10^{-3} to 10^{-4} for a faster convergence rate.
Instead, for the Feature extraction part, which is a common block of LINDEN and

SIERRA, a staircase-like variation is imposed as:

$$\alpha = \begin{cases} 10^{-3} \text{ for epochs} \leq 200 \\ 10^{-4} \text{ for } 200 \geq \text{epochs} \leq 400 \\ 5 \cdot 10^{-5} \text{ for } 400 \geq \text{epochs} \leq 500 \end{cases}$$

All the training and testing have been performed on Google Colaboratory (Colab). This free platform provides Python interactive notebooks with built-in libraries, e.g. Tensorflow, and high-end GPU (e.g. Nvidia Tesla K80) access speeding up the computation time for training NN.
That said, the GPU up-time is limited and, as Colab is web service with limited resources for the free subscription model.
Due to these issues, training has been performed considering sessions between 100 and 300 epochs. After 2 or 3 sessions the convergence is reached. The Training and Validation loss, the Training and Validation accuracy and the confusion matrix performed on the testing subset are computed.

5 | Results

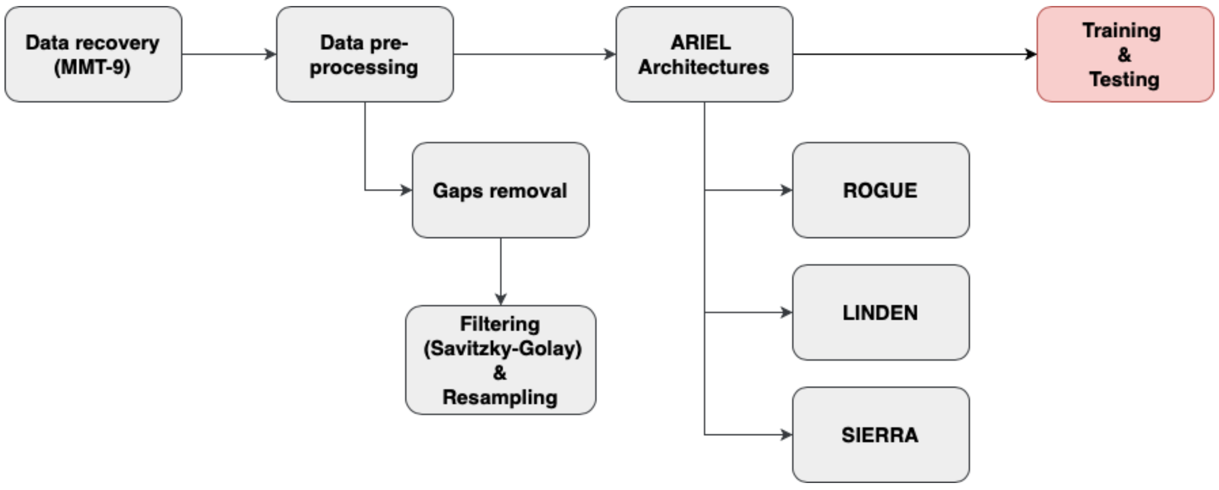


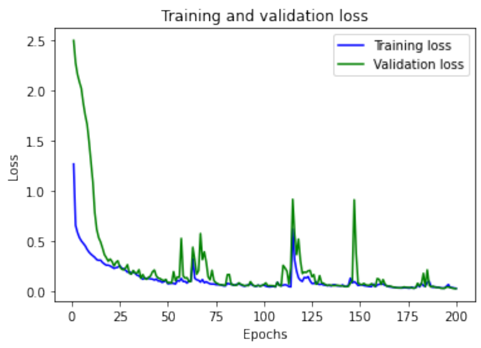
Figure 5.1: Thesis workflow - Results

This chapter illustrates the results obtained during the training and testing of the networks discussed in Chapter 4. The different case scenarios discussed in Section 4.4 are employed so as to underline ARIEL advantages and limitations.
A peculiar test case related to the identification of the Debris type, is also analysed in Section 5.4.

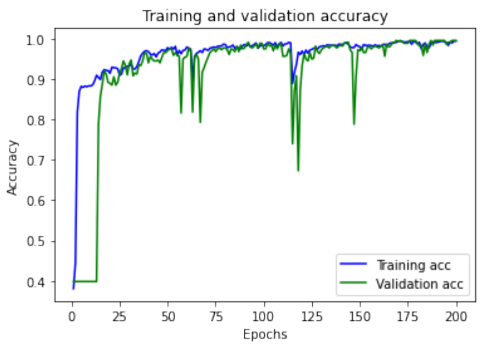
5.1. ROGUE - Results

Nominal conditions

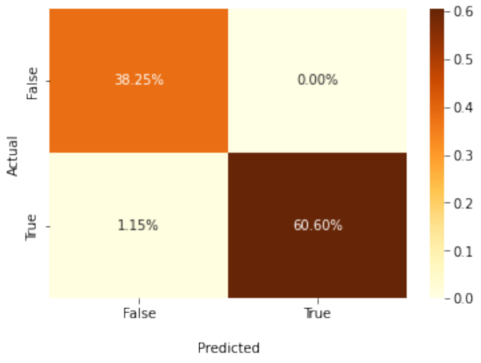
The first test considers the dataset Nom_1, which takes into account only periodic space objects. The corresponding results are shown in Figure 5.2.
The loss function quickly converges to a minimum value while the accuracy reaches almost 1 both for Training and Validation. This shows that the model considered is "just-right" with respect to the bias-variance trade-off. Moreover, the performances of the model are confirmed by the confusion matrix. In fact, as the off-diagonal terms clearly show, there are almost no false-positive predictions.
Afterwards, aperiodic spacecrafts are added, forming the Nom_2 dataset. This dataset



(a) Training and Validation loss



(b) Training and Validation accuracy

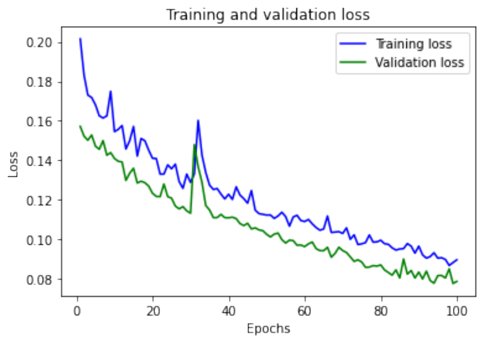


(c) Confusion matrix

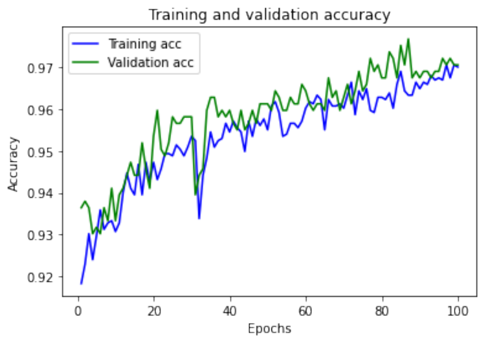
Figure 5.2: Performances for ROGUE on Nom_1

represents the expected nominal condition for ARIEL. Figure 5.3 displays the results obtained during the second training and testing session for this dataset.
In the first training segment the metrics reached are not comparable to the previous ones, therefore a second session was performed. It in fact allowed to significantly improve both the loss and the accuracy.
Some small peaks can still be observed both in loss and accuracy, probably due to the use of data batches during training, as they are limited to 2%. Moreover a slight underfitting can be observed, in particular in the loss function. This may be due to how the training processed those particular subsets considered for Training and Validation, after the shuffling of the overall Nom_2 dataset.
All things considered, the achieved performances are compliant to a reliable first classification of the light curve. This is once again confirmed by the confusion matrix, being the percentage of false-positives much lower (around 1%) than the diagonal terms, similarly to the training outcome with Nom_1 case.

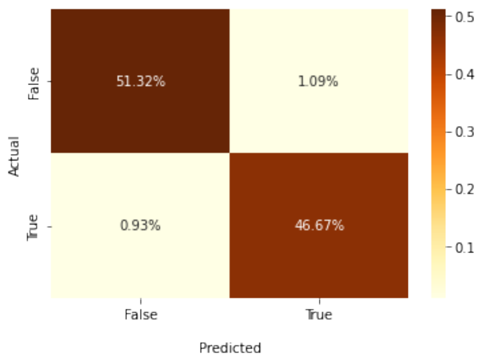
Variability test



(a) Training and Validation loss



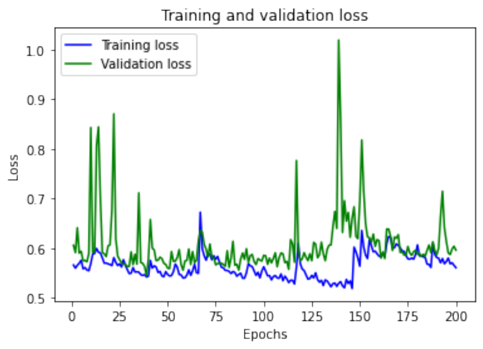
(b) Training and Validation accuracy



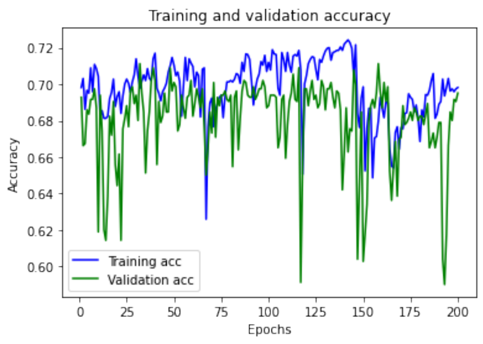
(c) Confusion matrix

Figure 5.3: Performances for ROGUE on Nom_2

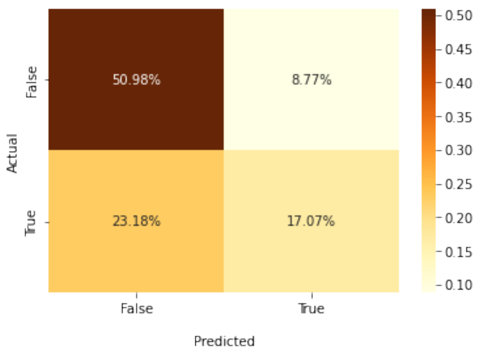
In these tests, the capabilities of the ROGUE network are challenged by considering the datasets in Section 4.4, featuring a higher variability in terms of the platforms analysed. First the Var_1 dataset is considered.
As shown in Figure 5.4, the performances drop as expected: even after training several times (after 2 training sessions of 200 epochs), not only does the loss and the accuracy not improve, but the model results in inaccurate predictions too. The confusion matrix clearly points this out.
This high variability in the dataset considered prevents the network to grasp fundamental characteristics from the light curves considered, therefore worse classification performance. This might be solved by trying to reduce the number of platforms in the dataset, e.g. from over 70 space objects in Var_1 to 40 in Var_2. The metrics are depicted in Figure 5.5.
These results are obtained after 7 training sessions of 200 epochs. In an attempt to improve model performance, it has been tested that no further increase can be achieved with more training sessions. Despite the intensive training phase, the metrics reached are still inferior to the Nominal conditions. Nonetheless, they can still be considered in line with what has been done in past works (cf. Section 1.2). The confusion observed is in



(a) Training and Validation loss



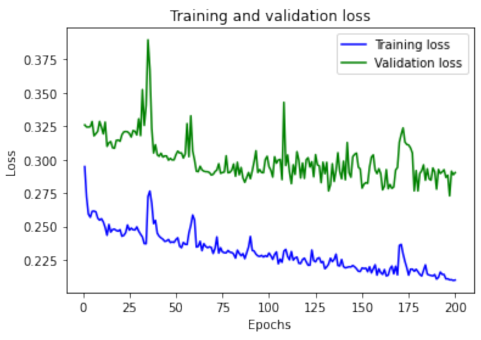
(b) Training and Validation accuracy



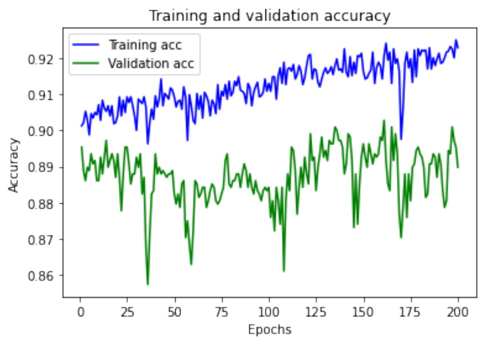
(c) Confusion matrix

Figure 5.4: Performances for ROGUE on Var_1

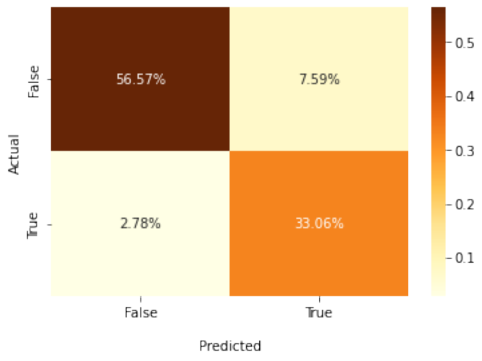
fact below the 10% threshold. It can be noted that the previously mentioned classes can still be observed in this confusion matrix, as Satellites and Debris are predominant with respect to Rocket bodies.
Moreover, significant overfitting is testified by the achieved performance metrics. Even if the Training and Validation loss functions tend to converge, the latter does not decrease as fast as the former. Regarding the accuracy, while the Training one is increasing, the Validation one remains somewhat constant. Therefore this difference is exacerbated as the training goes on, leading to the model eventually not progressing.
To reduce the overfitting observed, a further reduction in number of platforms is applied. Therefore the Var_3 dataset, which considers only 20 different platforms, is built and tested. The obtained results are shown in Figure 5.6.
Even in this case, the metrics obtained are lower than in the Nominal conditions, although the trend of increased performance with decreased platform is confirmed. A slight overfitting can still be observed: in fact more than 5 percentage points separate the Training and the Validation subset accuracy and an almost 10 percentage points gap is present in the loss.
Nevertheless, the confusion is still limited, remaining below 10%.



(a) Training and Validation loss



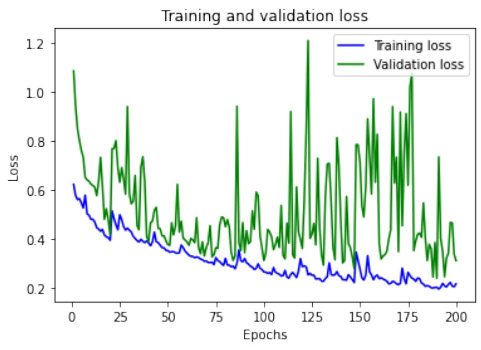
(b) Training and Validation accuracy



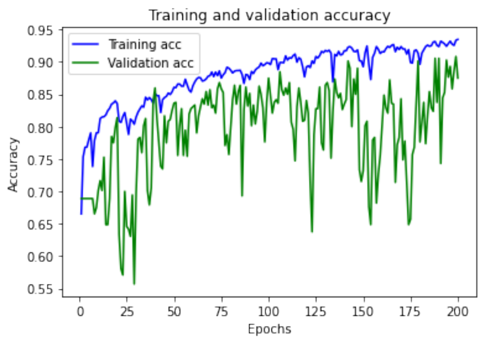
(c) Confusion matrix

Figure 5.5: Performances for ROGUE on Var_2

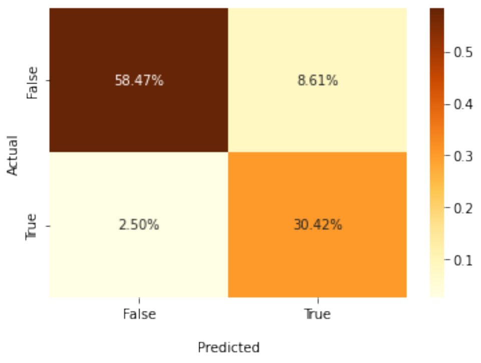
All the performances reached for ROGUE are summarized in Table 5.1.
Taking these results into consideration, it is reasonable to affirm that ROGUE is capable of distinguishing the Rocket body type among other types of spacecrafts. The combination of CNN and LSTM allowed to clearly distinguish when a light curve belongs to a Rocket body or not.
Hence the choice to use this network as a baseline for LINDEN is justified.



(a) Training and Validation loss



(b) Training and Validation accuracy



(c) Confusion matrix

Figure 5.6: Performances for ROGUE on Var_3

Dataset	**Loss reached**	**Accuracy reached**	**Note**
Nominal_1	~ 0.1	$\sim 98\%$	
Nominal_2	~ 0.1	$\sim 97\%$	Result obtained after 2 training portions of 200 then 100 epochs
Variability_1	$\geq 0.5; \leq 0.56$	$\geq 71\%; \leq 74\%$	Result obtained after 3 training portions of 200 epochs Overfitting starts to be prevalent
Variability_2	$\geq 0.23; \leq 0.3$	$\geq 89\%; \leq 92\%$	Result obtained after 8 training portions of 200 epochs Slight overfitting observed
Variability_3	$\geq 0.15; \leq 0.2$	$\geq 93\%; \leq 95\%$	Result obtained after 3 training portions, first of 100 epochs then two of 200 epochs Slight overfitting still observed

Table 5.1: Performances of ROGUE

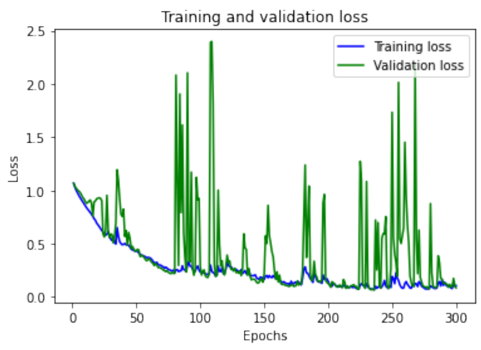
5.2. LINDEN - Results

LINDEN has the aim to predict the type of objects from two light curves which are then compared so as to establish if they are correlated or not. The first part is done by the Feature extraction block, which is then inserted in the overall Correlation block.
For this architecture, training had to be performed in two steps: first considering only the Feature extraction block and then the overall network. In the same way as ROGUE, the datasets mentioned in Section 4.4 are used for training and testing.

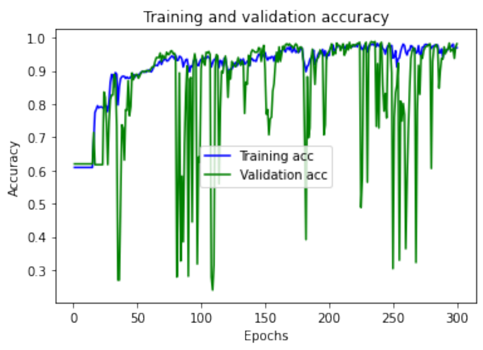
Feature extraction

Nominal conditions

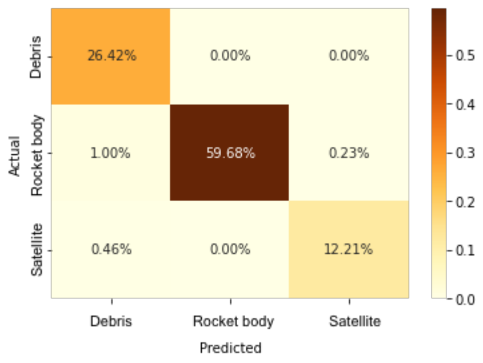
Starting from the Nominal conditions, both Nom_1 and Nom_2 are employed for training and testing. The results obtained are depicted in Figure 5.7 and 5.8 respectively.



(a) Training and Validation loss



(b) Training and Validation accuracy

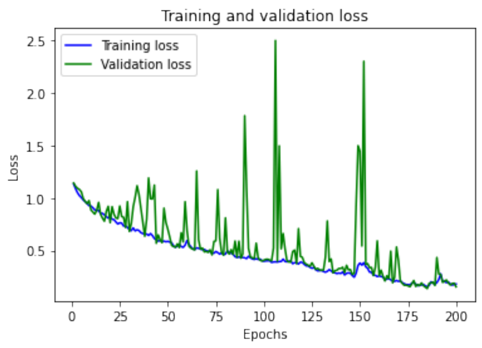


(c) Confusion matrix

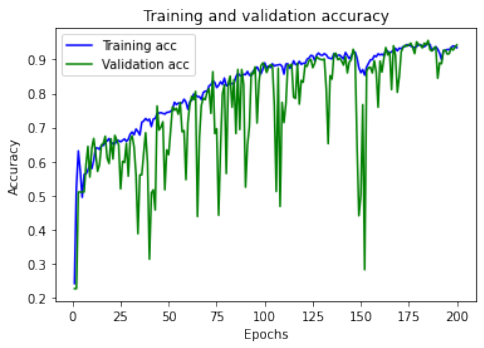
Figure 5.7: Performances for LINDEN Feature extraction on Nom_1

When trained with Nom_1, as only periodic spacecrafts are considered, the network properly distinguishes the categories involved and shows remarkable performances. In fact an almost certain prediction can be performed on this dataset, as the almost diagonal

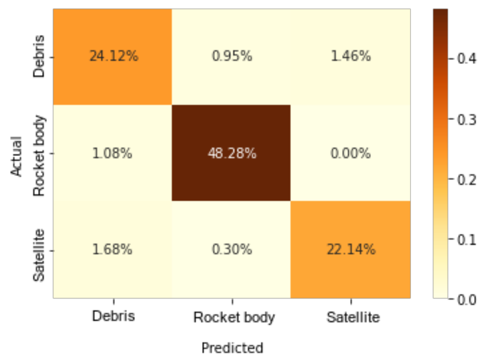
confusion matrix shows. It can also be observed that the Rocket body prevalence inside Nom_1 is also respected.
Many spikes are observed regarding the Validation performances: this may be linked to the batches used during training. In fact, since the Validation subset is much smaller with respect to the others, the batches considered for the minimisation process represent a bigger portion of that subset. This leads to having a noisy evaluation of the Validation loss.



(a) Training and Validation loss



(b) Training and Validation accuracy



(c) Confusion matrix

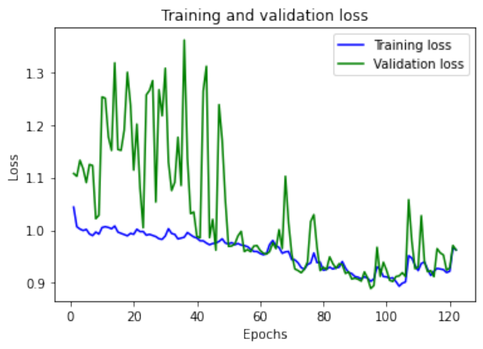
Figure 5.8: Performances for LINDEN Feature extraction on Nom_2

The results shown for Nom_2 confirm the performances expected by this kind of network. The levels reached for both loss and accuracy are in line with the ones achieved in the networks described in Section 1.2.
However spikes are still observed, probably due to the above-mentioned reasoning. The confusion is also bounded around 1% and the Rocket body type are still the most prevalent category observed, as expected.

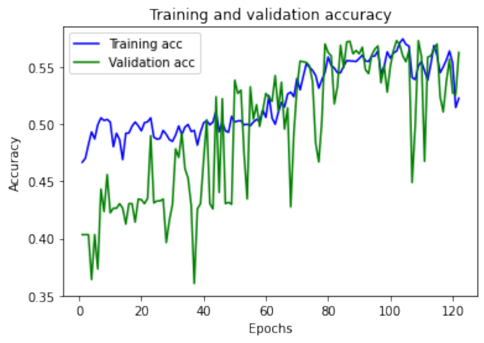
Variability test

Also for the Feature extraction block, the Variability datasets are used to assess the extent of this network capabilities. Figure 5.9, 5.10 and 5.11 show the obtained results from

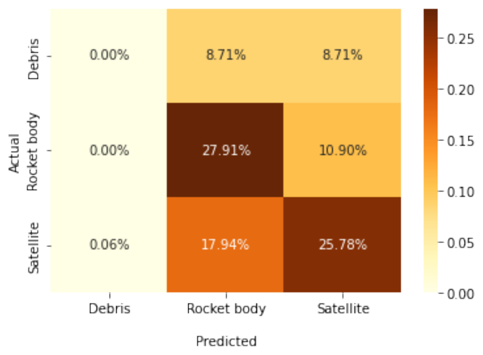
Var_1, Var_2 and Var_3 respectively.



(a) Training and Validation loss



(b) Training and Validation accuracy



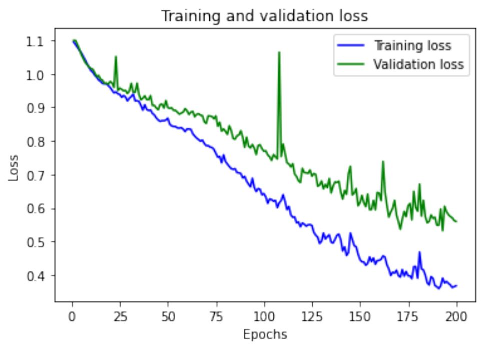
(c) Confusion matrix

Figure 5.9: Performances for LINDEN Feature extraction on Var_1

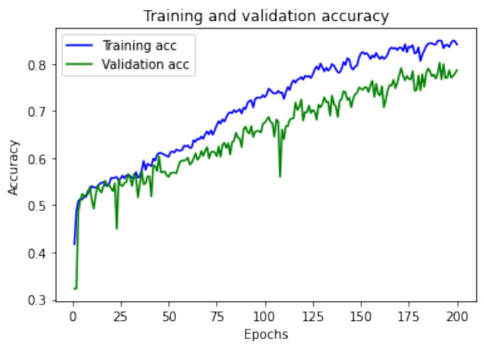
The Var_1 test proved to be the most challenging for LINDEN: in fact no improvements for the loss and the accuracy are observed and the Debris type is completely ignored. Many small modifications have been performed to the model and to the training methods. Regarding the model, the first modifications tried simplify the model through additional Dropout layers or higher regularizations terms. However, as the task posed is not trivial considering that Var_1 contains 70 different platforms, most probably the underlying problem is related to underfitting. Therefore additional CNN or LSTM layers have been considered, but still did not improve the situation. Regarding the training methods on the other hand, different optimizers, such as SGD, and smaller or larger learning rates have been tested. None of these modifications gave significant improvements.

Therefore this dataset was not tested in the Correlation evaluation part of LINDEN nor on SIERRA.

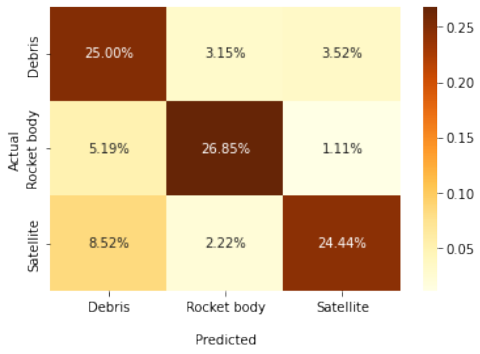
Using the Var_2 dataset improved significantly the performances of LINDEN. However strong overfitting can be observed: it creates a 20% difference between Training and Validation loss, and 10% gap between Training and Validation accuracy. This may be due to the complex task this network has to perform: it has to differentiate 40 different platforms



(a) Training and Validation loss



(b) Training and Validation accuracy

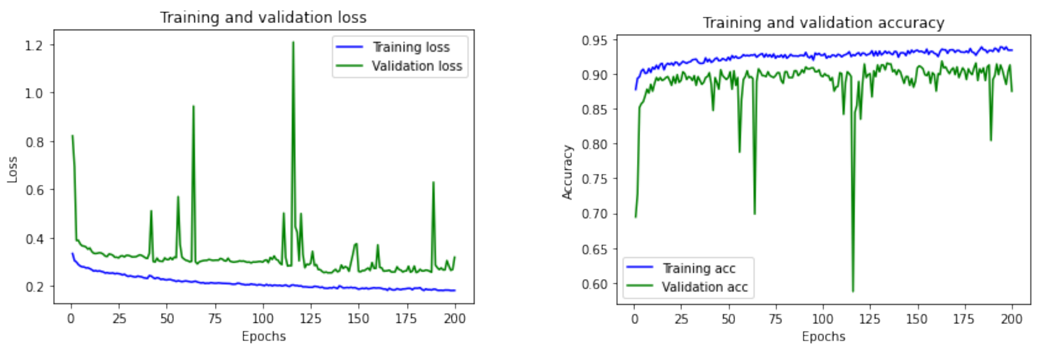


(c) Confusion matrix

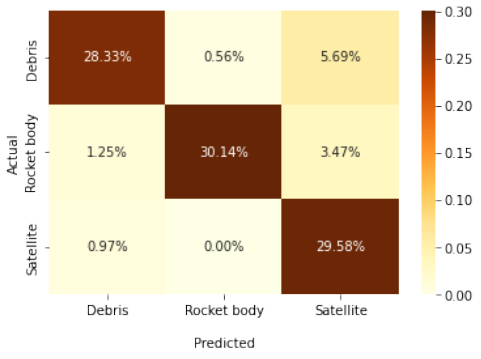
Figure 5.10: Performances for LINDEN Feature extraction on Var_2

which may have similar geometries or motion.
Despite this, the false-positive predictions are overall bounded below 10%. The proportions between types are also similar as a class weight is considered to properly balance out the dataset.
The results for the dataset Var_3 shown in Figure 5.11 correspond to the second training session of 200 epochs.
The performances strongly increased by inserting the Max Pooling layer in between the CNN and LSTM cell block. The overfitting is still noticeable however it is more limited than in the previous cases. In addition, the presence of false-positives is even less significant with respect to the previous cases, below 5%.

The overall performances for the Feature extraction block are thus summarized in Table 5.2.
Except for Var_1, all the results obtained from the datasets allow a proper identification of the type of object the light curve belongs to. As expected, a higher platform variability introduces more confusion. The overfitting is also exacerbated due to the increasing complexity the Variability datasets introduces.



(a) Training and Validation loss (b) Training and Validation accuracy



(c) Confusion matrix

Figure 5.11: Performances for LINDEN Feature extraction on Var_3

Worth to be noted is that the highest confusion occurs while considering the Debris type, as all the confusion matrices obtained prove. In fact these objects are either fragmentations outcomes, no longer functional satellites or rocket body parts. In the latter case the confusion with Satellites or Rocket bodies is more significant. Therefore it may be interesting to consider the Debris type separately, as done afterwards in Section 5.4.

Correlation block

It is worth mentioning that the dataset Var_1 has not been considered for the Correlation evaluation part of LINDEN, due to the limited results obtained in the Feature extraction block on this dataset.

Nominal conditions

The performances on Nom_1 and Nom_2 are depicted in Figure 5.12 and 5.13 respectively.
The accurate results obtained on the Feature extraction block on Nom_1 are reflected also in the Correlation block. Both the loss and the accuracy indicate the success of the

Dataset	Loss reached	Accuracy reached	Note
Nominal_1	~ 0.1	$\sim 98\%$	
Nominal_2	~ 0.1	$\sim 95\%$	
Variability_1	~ 1.0	$\geq 50\%; \leq 53\%$	Several modifications performed but no consequent improvements
Variability_2	$\geq 0.24; \leq 0.3$	$\geq 86\%; \leq 90\%$	Result obtained after 3 training portions of 200 epochs Relevant overfitting observed
Variability_3	$\geq 0.15; \leq 0.25$	$\geq 90\%; \leq 94\%$	Result obtained after 4 training portions, first two of 200 epochs then last two of 100 epochs Overfitting still observed

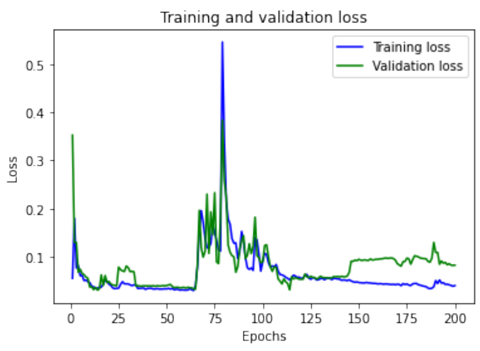
Table 5.2: Performances of LINDEN - Feature extraction

training which is also confirmed by the final confusion matrix.
A strange behaviour between around 70 and 110 epochs and a slight overfitting for the loss is observed, however it does not affect the overall performance.
For Nom_2, smaller portions of 50 epochs have been trained due to hardware limitations: in Figure 5.13 are shown the results of the last portion.
Even though the metrics obtained are still much of interest and the false-positive results are limited, overfitting yet represents an issue. Nevertheless, the gap between Training and Validation are not as significant for the accuracy than for the loss. This phenomenon however seems to rise independently from the results obtained in the Feature extraction block.

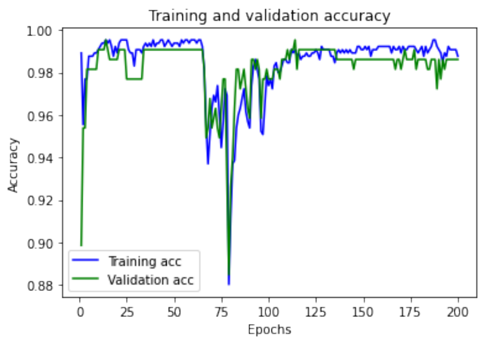
Variability test

As above-mentioned, the Var_1 is not considered for the training of this block, since the results obtained in the Feature extraction block do not allow a proper correlation evaluation. The results for Var_2 and Var_3 are depicted in Figure 5.14 and 5.15 respectively.

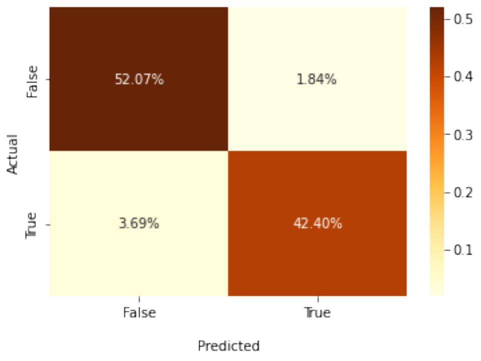
As the levels reached in the Feature extraction block leave room for uncertainty, the results obtained for Var_2 are much inferior. They also present a more substantial overfitting and the confusion reaches almost 15%.



(a) Training and Validation loss



(b) Training and Validation accuracy



(c) Confusion matrix

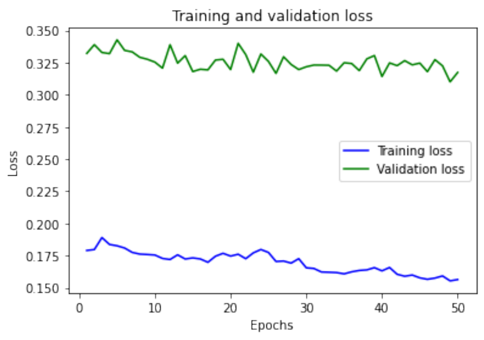
Figure 5.12: Performances for LINDEN Correlation evaluation on Nom_1

A slight improvement can be perceived for the performances reached using Var_3. The loss and the accuracy obtained are better than for Var_2 and the confusion observed is back to being below 10%.

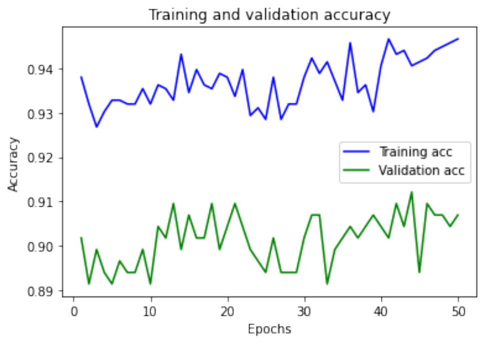
All in all, both Nominal and Variability test cases show promising results related to the Correlation evaluation. Table 5.3 outlines the above-mentioned metrics.

As expected whenever the Feature extraction block does not perform at its best, meaning that the predictions are not as accurate, this uncertainty gets propagated to the Correlation block. This is related mainly to the reasoning done in Section 4.2, explaining the Dot layer, specifically mentioning the intermediate case shown in Figure 4.5c.

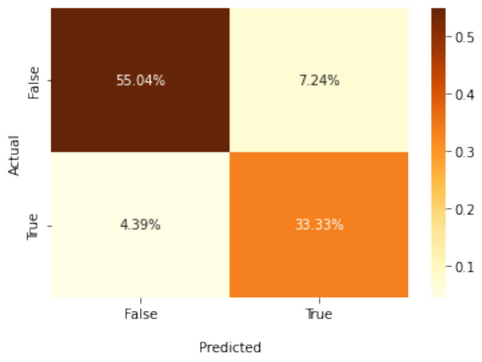
Therefore, improvements need to be considered for the Feature extraction block while analysing datasets with a high platform variability, so as to grant a much accurate evaluation in the Correlation block.



(a) Training and Validation loss



(b) Training and Validation accuracy

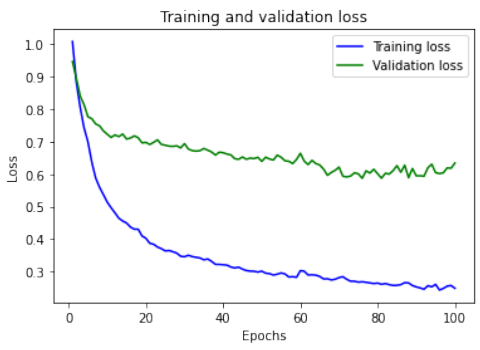


(c) Confusion matrix

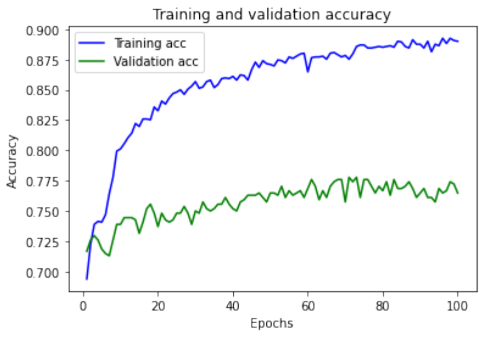
Figure 5.13: Performances for LINDEN Correlation evaluation on Nom_2

Dataset	**Loss reached**	**Accuracy reached**	**Note**
Nominal_1	$\geq 0.04; \leq 0.08$	$\sim 99\%$	Results obtained after 2 portions of 100 epochs
Nominal_2	$\geq 0.16; \leq 0.32$	$\geq 90\%; \leq 94\%$	Results obtained after 3 portions of 50 epochs Slight overfitting observed
Variability_2	$\geq 0.4; \leq 0.7$	$\geq 75\%; \leq 87\%$	Relevant overfitting observed
Variability_3	$\geq 0.25; \leq 0.35$	$\geq 87\%; \leq 90\%$	Result obtained after 2 training portions of 100 epochs Relevant overfitting still observed

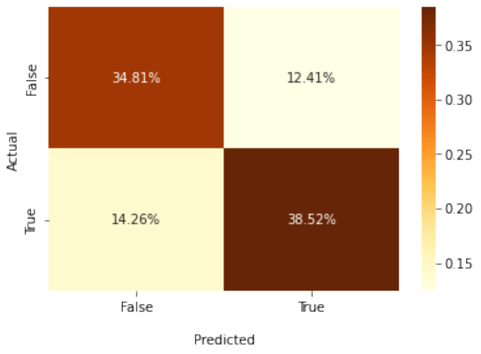
Table 5.3: Performances of LINDEN - Correlation evaluation



(a) Training and Validation loss

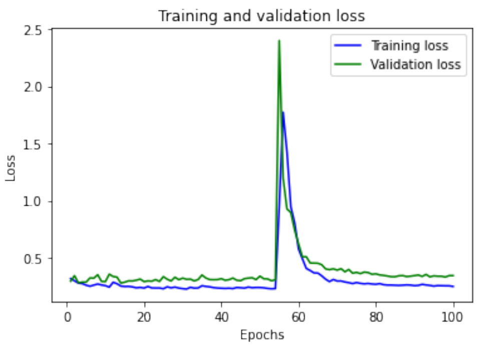


(b) Training and Validation accuracy

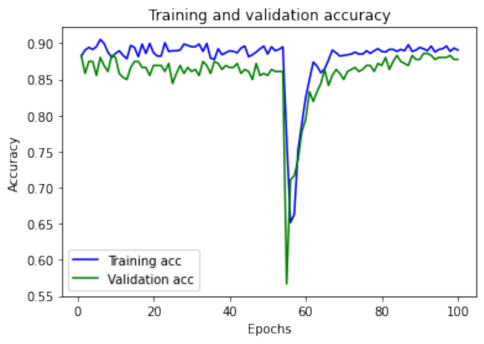


(c) Confusion matrix

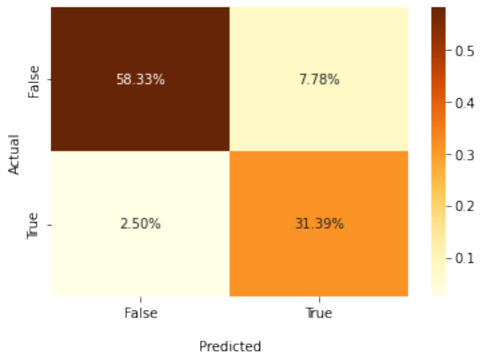
Figure 5.14: Performances for LINDEN Correlation evaluation on Var_2



(a) Training and Validation loss



(b) Training and Validation accuracy



(c) Confusion matrix

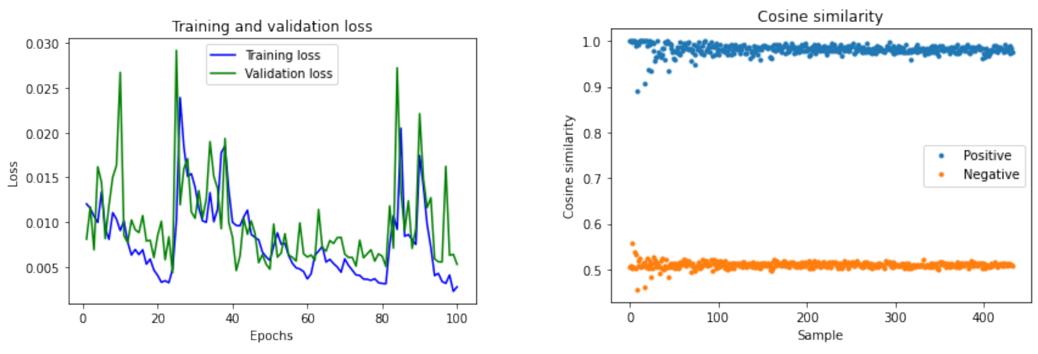
Figure 5.15: Performances for LINDEN Correlation evaluation on Var_3

5.3. SIERRA - Results

The Siamese network built in SIERRA employs the Feature extraction block of LINDEN as its embedding model (cf. Section 4.3).
Therefore, for this architecture as well, Var_1 dataset is not considered due to the poor results obtained in the Feature extraction block.
It is also important to keep in mind that the metrics considered for this architecture are the loss function and the Cosine Similarity. The latter in fact gives an indication on the embeddings performed in the Siamese model, by computing the angular distance between Anchor and Positive or Negative outcomes.

Nominal conditions

The metrics obtained for Nom_1 and Nom_2 datasets are shown in Figure 5.16 and 5.17.



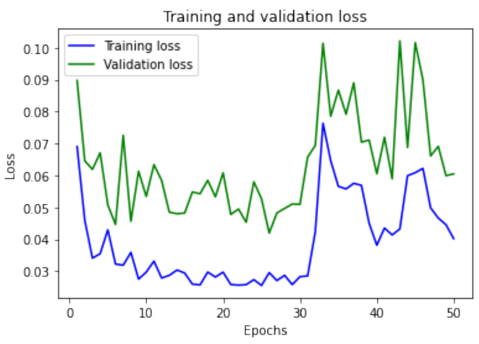
(a) Training and validation loss (b) Cosine similarity

Figure 5.16: Performances for SIERRA on Nom_1

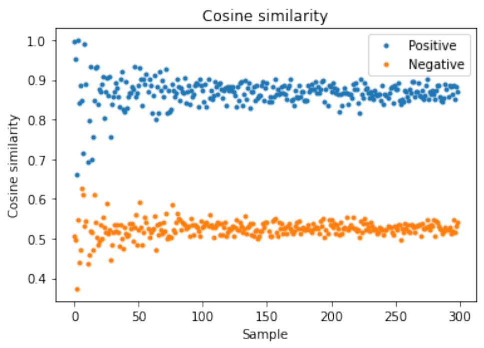
Regarding Nom_1, apart from the oscillating loss which however still remains contained around 0.1%, the results are still satisfactory. The clear separation in the similarity between Positive and Negative outcomes indicates the capability of the embedding model to distinguish similar types from different ones. This allows therefore to confirm the previous performances.
For Nom_2, the loss obtained is slightly higher than in the previous case, as expected. However the gap between Positive and Negative similarity is still as relevant, confirming the above-mentioned reasoning.

Variability test

Figure 5.18 and 5.19 illustrate the metrics obtained for Var_2 and Var_3 respectively.

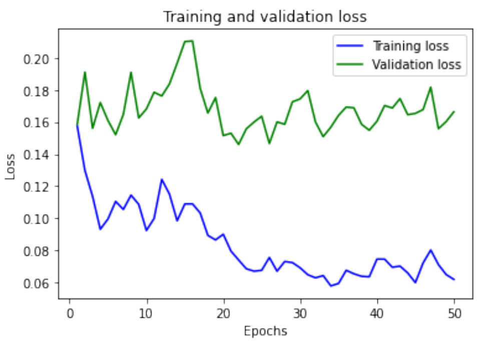


(a) Training and validation loss

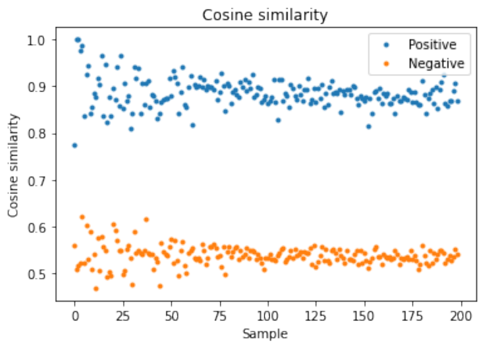


(b) Cosine similarity

Figure 5.17: Performances for SIERRA on Nom_2



(a) Training and validation loss

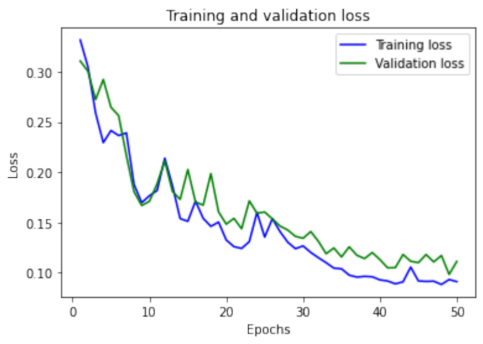


(b) Cosine similarity

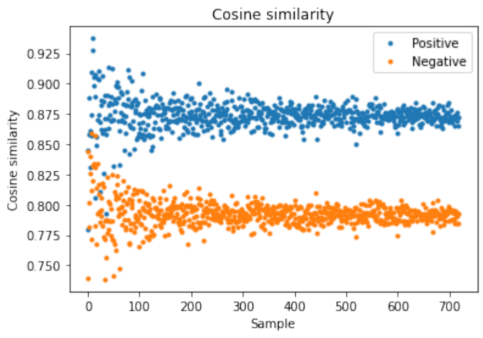
Figure 5.18: Performances for SIERRA on Var_2

As expected, the overfitting present in the Feature extraction block for Var_2 has propagated to SIERRA, giving a 10% difference between training and validation loss.
The Cosine Similarity between Positive and Negative is however still quite distinct meaning that little confusion is present. The points are more spread due to the higher uncertainty in the Feature extraction model for this dataset.
Even though the overfitting observed in the loss is smaller than in the previous case, the similarity for Var_3 is narrower in this test. This may be due to the fact that Anchor and Negative still have common characteristics, which can be mistaken for Positive.
In this case however the results obtained from the Feature extraction block are less uncertain, therefore the distributions of the points are denser.

As was done also for ROGUE and LINDEN, Table 5.4 shows the metrics for the datasets considered.
SIERRA is a valid alternative for LINDEN to evaluate the correlation between light curves. However it needs a reference, the Anchor, so as to perform this analysis. This



(a) Training and validation loss



(b) Cosine similarity

Figure 5.19: Performances for SIERRA on Var_3

Dataset	Loss reached	Similarity reached
Nominal_1	< 0.01	Positive: ~ 0.99 Negative: ~ 0.5
Nominal_2	$\geq 0.04; \leq 0.06$	Positive: ~ 0.89 Negative: ~ 0.52
Variability_2	$\geq 0.07; \leq 0.16$	Positive: ~ 0.89 Negative: ~ 0.55
Variability_3	$\geq 0.08; \leq 0.11$	Positive: ~ 0.87 Negative: ~ 0.78

Table 5.4: Performances of SIERRA

implies that an a-priori accurate guess on the type of spacecraft the light curve belongs to has to be performed, in order to properly choose the Positive and Negative outcome. This reasoning enforces the need to improve the Feature extraction block, in particular if many different platforms are considered.

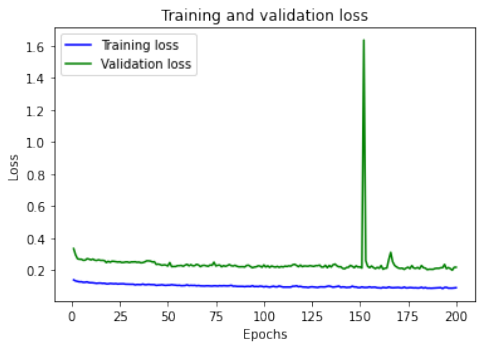
5.4. Debris test case

During the above-mentioned training, a particular factor can be considered: the Debris type might actually influence the classification done. In fact, these objects are either results of fragmentations or unused spacecrafts or rocket body parts. In the latter case, in particular, the resulting light curves are similar to the spacecrafts classified as Rocket bodies or Satellites. Thus the confusion might be resolved by eliminating this type during training and afterwards verify the origin of the Debris-type objects considered.
This procedure is performed only for LINDEN Feature extraction block, for the datasets described hereafter:

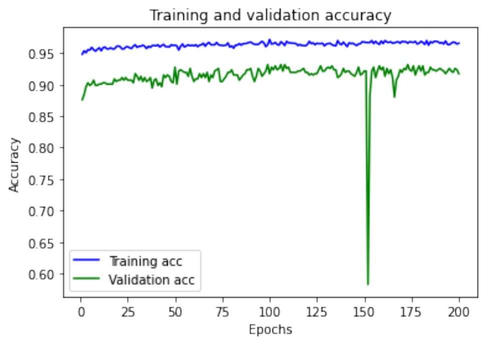
No_Debris_training (NoDeb_train) : It does not include Debris, resulting in 21 platforms: 7 Rocket bodies and 14 Satellites. It comprises 1200 light curves, "multiplied" 2 times. This dataset is used for Training and Validation.

Debris_test (Deb_test) : It only considers Debris, labeled as their origin - Rocket body or Satellite. Therefore 16 platforms are analysed: 7 Rocket bodies and 9 Satellites. This dataset includes 820 light curves and is employed for Testing only.

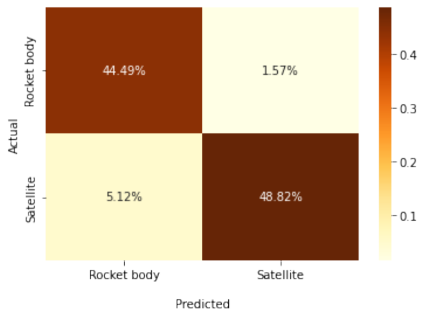
In these datasets, both periodic and aperiodic spacecrafts are considered.
Thus the training is performed considering the NoDeb_train dataset: the results are shown in Figure 5.20.
These results are obtained after the second portion of 200 training epochs. Even though the overfitting is still present, the performances reached are higher than the Variability test cases'.
This model is then tested using the Debris_test database. The resulting confusion matrix is shown in Figure 5.21.
This confusion matrix indicates that the Feature extraction model still requires some training to correctly pin-point the origin of Debris.
However, it might be the case that in Deb_test fragmentation-caused debris are included as well, being difficult to relate to the original object, thus complicating type identification.
Further improvements could entail an additional classification label as "Fragmentation debris" which accounts for this case.



(a) Training and Validation loss



(b) Training and Validation accuracy



(c) Confusion matrix

Figure 5.20: Performances for LINDEN Feature extraction on NoDeb_train

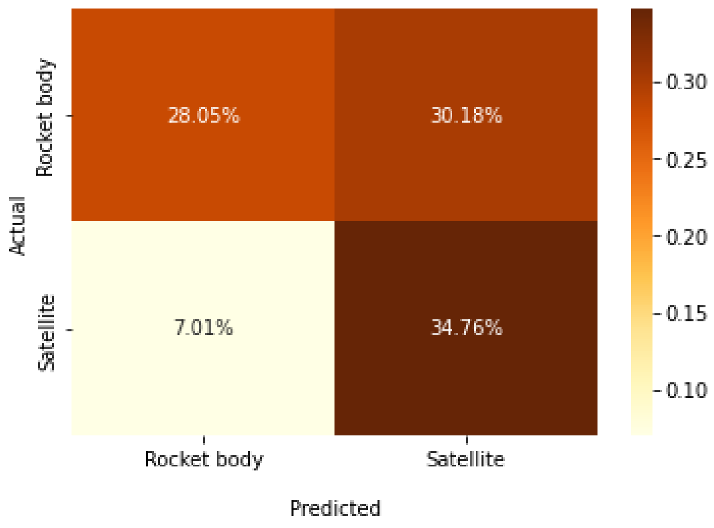


Figure 5.21: Confusion matrix on Deb_test

6 | Conclusions and future developments

As the issue of the ever-growing space population is gaining more and more importance in the space sector, it is of paramount importance to recognise the objects observed during surveys.
ARIEL provides a strategy to identify objects according to their type and to establish a degree of correlation between the unknown object and a catalogued one. This is done through the analysis of light curves employing machine learning-based systems. The NN considered combines the capabilities of CNN and of LSTM: the former allows to extract general characteristics while the latter focuses on time-dependent features.
Three architectures are thus proposed, each focusing on a different aspect of the problem at-hand:

- ROGUE recognises the Rocket body type among a pool of space objects
- LINDEN first classifies in parallel two light curves according to the spacecraft type; subsequently evaluates if they belong to the same category
- SIERRA adapts a Siamese network to evaluate the distances between a target object light curve and two alternative categories

The light curves are obtained from the MMT-9 database and have been pre-processed by removing gaps and filtering with the SavGol smoothing filter.
After extensive training using different datasets, the performances have been assessed. In particular it can be observed that the loss convergence is attained below 0.5 and the accuracy reaches around 90%. The false positives also are generally bounded around 10% of the predictions. This fact is also confirmed by the significant gap that can be observed for the similarity in SIERRA.
However these NN generalization capability is limited to a small number of platforms. In fact, far too variable datasets do not allow an accurate type recognition, thus preventing the correlation evaluation to be performed.
Moreover, overfitting is occurs quite often: in some cases it becomes substantial, therefore

impacting the accuracy of the inference on a realistically distributed scenario.

Some options can hence be proposed to improve ARIEL:

- Consider a dataset with a small number of platforms, e.g. at maximum 20 overall
- Modify the strategies employed altogether, e.g. consider bigger networks, modify activation functions, types of layers, etc.
- Employ stricter regularization to reduce overfitting
- Restrict the problem to the recognition of platforms among a same type, or a same attitude regime (i.e. periodic, aperiodic or non variable)
- Focus the problem on the Debris type recognition: in fact, this type is actually the most confused as some of these objects are unused satellites or intact rocket body parts. Therefore, a dedicated analysis among Debris may be needed if those objects are involved.

Bibliography

[1] Inter-Agency Space Debris Coordination Committee, "Space debris mitigation guidelines," 2002.

[2] ESA Space Debris Office, "ESA annual space environment report," Tech. Rep. 6, European Space Agency, Darmstadt, Germany, April 2022.

[3] ESA Space Debris Office, "Space debris by the numbers," 2022. https://www.esa.int/Space_Safety/Space_Debris/Space_debris_by_the_numbers.

[4] ESA, "Sentinel-1 impact," 2022. https://www.esa.int/ESA_Multimedia/Images/2016/08/Sentinel-1_impact#.Y053J6kZKis.link.

[5] D. Kessler and B. Cour-Palais, *Collision frequency of artificial satellites: The creation of a debris belt*, pp. 2637–2646. Journal of Geophysical Research, 1978.

[6] ESA, "Clearspace-1," 2020. https://www.esa.int/Space_Safety/ClearSpace-1.

[7] ESA, "Sail solutions for space junk," 2020. https://www.esa.int/Enabling_Support/Space_Engineering_Technology/Shaping_the_Future/Sail_solutions_for_space_junk.

[8] EU SST, "European space surveillance and tracking," 2016. https://www.eusst.eu.

[9] A. Morselli, *High order methods for Space Situational Awareness.* PhD thesis, Politecnico di Milano, 2014.

[10] J. Silha, *Reviews in Frontiers of Modern Astrophysics: From Space Debris to Cosmology*, ch. 1, pp. 1–21. Springer Nature, 2020.

[11] B. Bradley and P. Axelrad, "Lightcurve inversion for shape estimation of GEO objects from space-based sensors," *ISSFD*, 2014.

[12] M. Ashikmin and P. Shirley, "An anisotropic Phong light reflection model," tech. rep., University of Utah, Salt Lake City, Utah, USA, 2000.

[13] R. Linares and R. Furfaro, "Space object classification using deep convolutional neural networks," *IEEE*, 2016.

[14] R. Furfaro, R. Linares, and V. Reddy, "Shape identification of space objects via lightcurve inversion using deep learning models," *AMOS*, 2019.

[15] K. McNally, D. Ramirez, A. M. Anton, D. Smith, and J. Dick, "Artificial intelligence for Space Resident Objects characterisation with lightcurves," (Darmstadt, Germany), ESA, ESA Space Debris Office, 4 2021. Proc. 8th European Conference on Space Debris.

[16] E. Kerr, G. Falco, N. Maric, D. Petit, P. Talon, E. Geistere Petersen, C. Dorn, S. Eves, N. Sánchez-Ortiz, R. Dominguez Gonzalez, and J. Nomen-Torres, "Light curves for GEO object characterisation," (Darmstadt, Germany), ESA, ESA Space Debris Office, 4 2021. Proc. 8th European Conference on Space Debris.

[17] NASA, "Electromagnetic transmittance, or opacity, of the earth's atmosphere," 2008. `https://commons.wikimedia.org/wiki/File:Atmospheric_electromagnetic_opacity.svg`.

[18] H. Karttunen, P. Kröger, H. Oja, M. Poutanen, and K. Donner, *Fundamental Astronomy*. Springer Science & Business Media, 4 ed., 2003.

[19] M. Massari, *Optical instruments*. DAER, Politecnico di Milano, 2021. Lecture presentation of Payload Design course.

[20] S. Karpov, E. Katkova, G. Beskin, A. Biryukov, S. Bondar, E. Davydov, E. Ivanov, A. Perkov, and V. Sasyuk, "Massive photometry of low altitude artificial satellites on Mini-Mega Tortora," 2015. Database: `http://mmt.favor2.info/satellites`.

[21] G. Beskin, S. K. A. Biryukov, S. Bondar, E. Ivanov, E. Katkova, N. Orekhova, A. Perkov, and V. Sasyuk, "Wide-field optical monitoring with Mini-Mega Tortora (MMT-9) multichannel high temporal resolution telescope," 2017.

[22] A. Savitzky and M. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Analytical Chemistry*, pp. 1627–1639, 1964. `https://doi.org/10.1021/ac60214a047`.

[23] N. Gallagher, "Savitzky-golay smoothing and differentiation filter," 2020. `https://eigenvector.com/wp-content/uploads/2020/01/SavitzkyGolay.pdf`.

[24] F. Bre, J. Gimenez, and V. Facchinotti, "Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks," *Energy and Buildings*, 2017.

[25] R. Cipollone and A. De Vittori, "Machine learning fundamentals," Politecnico di Milano, 3 2020.

[26] D. Kaidanovic, “Ai-aided optical navigation about uncooperative spacecraft using synthetic imagery,” Master’s thesis, Politecnico di Milano, 4 2020.

[27] V. Rajput, “Pooling layers in neural nets and their variants,” 2022. https://medium.com/aiguys/pooling-layers-in-neural-nets-and-their-variants-f6129fc4628b.

[28] S. Raschka, *Introduction to Recurrent Neural Networks.* Statistics Department, UW-Madison, 2021. Lectures from "Introduction to Deep Learning", https://sebastianraschka.com/blog/2021/dl-course.html.

[29] N. Backpropagation, “Neural network with backpropagation.” https://github.com/Vercaca/NN-Backpropagation.

[30] Y. Yin, *Deep Learning Notes.* Statistics Department, Columbia University, 2 2018. Based on deep learning course taught by Prof. Andrew Ng from Standford University. The course can be found in Coursera: https://www.coursera.org/learn/deep-neural-network.

[31] S.Singh, “Understanding the bias-variance tradeoff,” 2018. https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229.

[32] H. Essam and S. Valdarrama, “Image similarity estimation using a Siamese Network with a Triplet loss,” 2021. https://keras.io/examples/vision/siamese_network/.

[33] F. Schoff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” tech. rep., Google Inc., 2015. https://arxiv.org/pdf/1503.03832.pdf.

[34] US Department of Defence, “Database of satellite orbital parameters.” http://www.space-track.org.

[35] M. McCants, “Satellite tracking TLE page.” http://www.mmccants.org.

[36] CelesTrack, “Online satellite catalogue.” https://celestrak.org.

A | Appendix

A.1. Orbital regimes

Table A.1 summarizes the orbit classes definition according to ESA [2]. The parameters mentioned are semi-major axis a [km], eccentricity e, inclination i [deg], perigee height h_P [km] and apogee height h_A [km].

Orbit	Description	Definition
GEO	Geostationary Orbit	$i = [0; 25]$, $h_P = [35586; 35986]$, $h_A = [35586; 35986]$
IGO	Inclined Geosynchronous Orbit	$a = [37948; 48380]$, $e = [0.00; 0.25]$, $i = [25; 180]$
EGO	Extended Geosynchronous Orbit	$a = [37948; 48380]$, $e = [0.00; 0.25]$, $i = [0; 25]$
NSO	Navigation Satellites Orbit	$i = [50; 70]$, $h_P = [18100; 24300]$, $h_A = [18100; 24300]$
GTO	GEO Transfer Orbit	$i = [0; 90]$, $h_P = [0; 2000]$, $h_A = [31570; 40002]$
MEO	Medium Earth Orbit	$h_P = [2000; 31570]$, $h_A = [2000; 31570]$
LEO	Low Earth Orbit	$h_P = [0; 2000]$, $h_A = [0; 2000]$
MGO	MEO-GEO Crossing Orbits	$h_P = [2000; 31570]$, $h_A = [31570; 40002]$
HEO	Highly Eccentric Earth Orbit	$h_P = [0; 31570]$, $h_A > 40002$
LMO	LEO-MEO Crossing Orbits	$h_P = [0; 2000]$, $h_A = [2000; 31570]$

Table A.1: Orbital classes [2]

A.2. Data distribution from MMT database

Hereafter are listed the numbers of objects in each of the following categories:

- Orbital regime (cf. Table A.1)
- Attitude regime: periodic, aperiodic and non variable
- Type of object: Debris, Rocket body and Satellite
- Number of light curves per object: more than 50, less than 10 and in between

The following tables also show the various combination.

Orbit	**Number**
GEO	174
IGO	4
EGO	107
NSO	113
GTO	236
MEO	52
LEO	5206
MGO	53
HEO	37
LMO	332

Table A.2: Number of objects per orbit

Type of object	**Number**
Rocket bodies	827
Debris	839
Satellites	4648

Table A.3: Number of objects per type

Attitude regime	**Number**
Periodic	985
Aperiodic	1550
Non variable	3779

Table A.4: Number of objects per variability

Tracks	**Number**
Greater than 50	870
In between	1896
Less than 10	3548

Table A.5: Number of objects per number of light curves available

Type of object	Attitude regime		
	Periodic	Aperiodic	Non variable
Rocket bodies	379	264	184
Debris	267	215	357
Satellites	339	1071	3238

Table A.6: Number of objects per type and corresponding variability

Type of object	Tracks		
	Greater than 50	In between	Less than 10
Rocket bodies	249	290	288
Debris	112	213	514
Satellites	509	1393	2746

Table A.7: Number of objects per type and corresponding number of light curves available

Attitude regime	Tracks		
	Greater than 50	In between	Less than 10
Periodic	237	358	390
Aperiodic	514	547	489
Non variable	119	991	2669

Table A.8: Number of objects per variability and corresponding number of light curves available

Orbit	Type of object		
	Rocket bodies	Debris	Satellites
GEO	0	1	173
IGO	0	0	4
EGO	1	0	106
NSO	15	0	98
GTO	150	41	45
MEO	5	3	44
LEO	428	705	4073
MGO	33	9	11
HEO	19	10	8
LMO	176	70	86

Table A.9: Number of objects with a certain type per orbit

Orbit	Attitude regime		
	Periodic	Aperiodic	Non variable
GEO	22	68	84
IGO	0	0	4
EGO	13	37	57
NSO	29	7	77
GTO	163	31	42
MEO	12	17	23
LEO	530	1269	3407
MGO	35	7	11
HEO	26	3	8
LMO	155	111	66

Table A.10: Number of objects with a certain variability per orbit

Orbit	**Tracks**		
	Greater than 50	**In between**	**Less than 10**
GEO	4	60	110
IGO	0	0	4
EGO	0	13	94
NSO	7	17	89
GTO	12	69	155
MEO	14	9	29
LEO	730	1588	2888
MGO	5	21	27
HEO	0	13	24
LMO	98	106	128

Table A.11: Number of objects per orbit given their light curves available

List of Figures

List of Tables

List of Symbols

Light curves

Variable	Description	SI unit
λ	Wavelength	m
F, f/n	Aperture ratio	
D	Aperture	m
f	Focal length	m
θ	Resolution	
L	Intrinsic luminosity	W
l	Apparent luminosity	$\mathrm{W} \cdot \mathrm{m}^{-2}$
m	Apparent magnitude	
d, d_0	Distances	pc (parsec)
M	Absolute magnitude	
w	Window width (SavGol)	s
o	Polynomial order (SavGol)	

Deep Learning

Variable	Description	SI unit
$\mathbf{x}$	Input vector	
$\mathbf{y}$	Output label	
$\mathbf{W}$	Weight matrix	
$\mathbf{b}$	Bias vector	
$\mathbf{z}$	Linear combination of input vector	
$f(\cdot)$	Activation function	
$\mathbf{a}$	Activation output	
n	Input matrix dimension	
k	Kernel size	
s	Stride	
p	Padding	
C	LSTM cell state	
i	LSTM input node	
g	LSTM input gate	
f	LSTM forget gate	
o	LSTM output gate	
J	Cost function	
α	Learning rate	s^{-1}
λ	Regularization parameter	
$R(\mathbf{W})$	Regularization function	

Acknowledgements

Vorrei prima di tutto ringraziare il mio relatore, Prof. P. Di Lizia, per avermi proposto questo progetto e seguito sempre con grande interesse per tutta la sua durata. Un ringraziamento anche ai collaboratori che sono intervenuti, Prof M. Massari, A. De Vittori, R. Cipollone e L. Facchini, che mi hanno guidato durante tutto il lavoro di tesi, fornendo preziosi consigli e spunti di riflessione e per la loro costante disponibilità.

Vorrei inoltre particolarmente ringraziare la mia famiglia: grazie a papà Luca e a mamma Silvia per il loro incondizionato supporto e pazienza durante questi ultimi sei anni. Grazie anche a mia sorella Michela che nonostante la lontananza, la sua complicità era sempre con me. Vorrei ringraziare anche i miei nonni Ugo e Marisa, per tutte le candeline che hanno accesso per i miei esami (sperando non sia solo per quello che sono arrivata fin qui) e per la loro continua curiosità sul mio percorso. Un ringraziamento anche a mia nonna Silvana per il suo costante sostegno.
Un gros gros merci aussi à ma meilleure amie Jade, désormais ma soeur par adoption, qui a toujours été là pour moi et m'a toujours encouragée à croire en moi-même. Pour les vacances, les appels interminables, les petits dramas qui s'ensuivent, les rires et pleins pleins d'autres choses, merci tata.
Vorrei ringraziare anche Mary, Colla, Tom, Alex, Dinho e Jac per aver condiviso la non-trascurabile difficoltà della vita al Politecnico con tutte le lezioni, pause, esami, progetti e chi ne ha più ne metta, rendendo Bovisa sicuramente meno triste di quanto non sia. Un ringraziamento va anche ad Alle e a Fede su cui ho sempre potuto contare, per la loro gentilezza e spontaneità nonostante i diversi percorsi intrapresi. Grazie anche a tutte le persone che ho incontrato durante questi sei anni, ognuna ha apportato un suo contributo a questo percorso. Ringrazio anche Carola per aver condiviso con me l'esperienza del FLAM a cui sono grata per l'opportunità data.

Un ultimo ringraziamento va alla persona senza il quale molto probabilmente questo progetto non ci sarebbe: non solo per gli innumerevoli consigli, discussioni sul tema e l'estrema disponibilità in tutti gli aspetti ma anche per l'assoluto sostegno su tutti i fronti, che mi spinge ogni giorno ad essere una versione migliore di me. Quindi, grazie tato.