# Classification of Maritime Vessels using Convolutional Neural Networks

Mathias Anneken[1*], Moritz Strenger[1*], Sebastian Robert[2], and Jürgen Beyerer[12]

[1] Vision and Fusion Laboratory (IES), Karlsruhe Institute of Technology (KIT)
[*] *equal contribution, alphabetical order*
mathias.anneken@kit.edu, moritz.strenger@alumni.kit.edu
[2] Fraunhofer IOSB, Karlsruhe; Fraunhofer Center for Machine Learning
sebastian.robert@iosb.fraunhofer.de, juergen.beyerer@iosb.fraunhofer.de

**Abstract** Due to a steady increase in traffic at sea, the need for support in surveillance task is growing for coast guards and other law enforcement units all over the world. An important cornerstone is a reliable vessel classification, which can be used for detecting criminal activities like illegal, unreported and unregulated fishing or smuggling operations. As many ships are required to transmit their position by using the automatic identification system (AIS), it is possible to generate a large dataset containing information on the world wide traffic. This dataset is used for implementing deep neural networks based on residual neural networks for classifying the most common shiptypes based on their movement patterns and geographical features. This method is able to reach a competitive result. Further, the results show the effectiveness of residual networks in time-series classification.

**Keywords:** residual neural network, time series classificaiton, convolutional neural networks, maritime domain, ship classification

## 1 Introduction

In 2019 the world's commercial fleet consists of 95,402 ships with a total capacity of 1,976,491 thousand dwt. (a plus of 2.6 % in carrying capacity compared to last year) [1]. According to the International Chamber of Shipping, the shipping industry is responsible for about 90 % of all trade [2]. In order to ensure the safe voyage of all participant in the international travel at sea, the need for monitoring is steadily increasing.

While more and more data regarding the sea traffic is collected by using cheaper and more powerful sensors, the data still needs to be processed and understood by human operators. In order to support the operators, reliable anomaly detection and situation recognition systems are needed. One cornerstone for this development is a reliable automatic classification of vessels at sea.

For example by classifying the behaviour of non cooperative vessels in ecological protected areas, the identification of illegal, unreported and unregulated (IUU) fishing activities is possible. IUU fishing is in some areas of the world a major problem, e. g., »in the wider-Caribbean, Western Central Atlantic region, IUU fishing compares to 20-30 percent of the legitimate landings of fish« [3] resulting in an estimated value between USD 700 and 930 million per year.

One approach for gathering information on the sea traffic is based on the automatic identification system (AIS)[3]. It was introduced as a collision avoidance system. As each

[3] https://gpsd.gitlab.io/gpsd/AIVDM.html

vessel is broadcasting its information on an open channel, the data is often used for other purposes, like training and validating of machine learning models.

AIS provides dynamic data like position, speed and course over ground, static data like MMSI[4], shiptype and length, and voyage related data like draught, type of cargo, and destination about a vessel.

The system is self-reporting, it has no strong verification of transmission, and many of the fields in each message are set by hand. Therefore, the data can not be fully trusted. As Harati-Mokhtari et al. [4] stated, half of all AIS messages contain some erroneous data. As for this work, the dataset is collected by using the AIS stream provided by AISHub[5], the dataset is likely to have some amount of false data. While most of the errors will have no further consequences, minor coordinate inaccuracies or wrong vessel dimensions are irrelevant, some false information in vessel information can have an impact on the model performance.

## 2 Related Work

Classification of maritime trajectories and the detection of anomalies is a challenging problem, e.g., since classifications should be based on short observation periods, only limited information is available for vessel identification. Riveiro et al. [5] give a survey on anomaly detection at sea, where shiptype classification is a subtype.

Jiang et al. [6] present a novel TrajectoryNet capable of point-based classification. Their approach is based on the usage of embedding GPS coordinates into a new feature space. The classification itself is accomplished using an long short-term memory (LSTM) network.

Further, Jiang et al. [7] propose a partition-wise LSTM (PLSTM) for point-based binary classification of AIS trajectories into fishing or non-fishing activity. They evaluated their model against other recurrent neural networks and achieve a significantly better result than common recurrent network architectures based on LSTM or gated recurrent units.

A recurrent neural network is used by Nguyen et al. in [8] to reconstruct incomplete trajectories, detect anomalies in the traffic data and identify the real type of a vessel. They are embedding the position data to generate a new representation as input for the neural network.

Besides these neural network based approaches, other methods are also used for situation recognition tasks in the maritime domain. Especially expert-knowledge based systems are used frequently, as illegal or at least suspicious behaviour is not recorded as often as desirable for deep learning approaches.

Conditional Random Fields are used by Hu et al. [9] for the identification of fishing activities from AIS data. The data has been labelled by an expert and contains only longliner fisher boats.

Saini et al. [10] propose an hidden Markov model (HMM) based approach to the classification of trajectories. They combine Global-HMM and Segmental-HMM using a genetic algorithm. In addition, they tested the robustness of the framework by adding Gaussian noise.

In [11] Fischer et al. introduce a holistic approach for situation analysis based on Situation-Specific Dynamic Bayesian Networks (SSDBN). This includes the modelling of the SSDBN as well as the presentation to end-users. For a Bayesian Network, the

[4] "Maritime Mobile Service Identity" as unique identifier for the association of data

[5] https://www.aishub.net

parametrisation of the conditional probability tables is crucial. Fischer introduces an algorithm for choosing these parameters in a more transparent way. Important for the functionality is the ability of the network to model the domain knowledge and the handling of noisy input data. For the evaluation, simulated and real data is used to assess the detection quality of the SSDBN.

Based on DBNs, Anneken et al. [12] implemented an algorithm for detecting illegal diving activities in the North Sea. As explained by de Rosa et al. [13] an additional layer for modelling the reliability of different sensor sources is added to the DBN.

## 3 Preprocessing

In order to use the AIS data, preprocessing is necessary. This includes cleaning wrong data, filtering data, segmentation, and calculation of additional features. The whole workflow is depicted in Figure 1. The input in form of AIS data and different maps is shown as blue boxes. All relevant MMSIs are extracted from the AIS data. For each MMSI, the position data is used for further processing. Segmentation into separate trajectories is the next step (yellow). The resulting trajectories are filtered (orange). Based on the remaining trajectories, geographic (green) and trajectory (purple) based features are derived. For each of the resulting sequences, the data is normalized (red), which results in the final dataset. Only the 6 major shiptypes in the dataset are used for the evaluation. These are “Cargo”, “Tanker”, “Fishing”, “Passenger”, “Pleasure Craft” and “Tug”. Due to their similar behaviour, “Cargo” and “Tanker” will combined to a single class “Cargo-Tanker”.

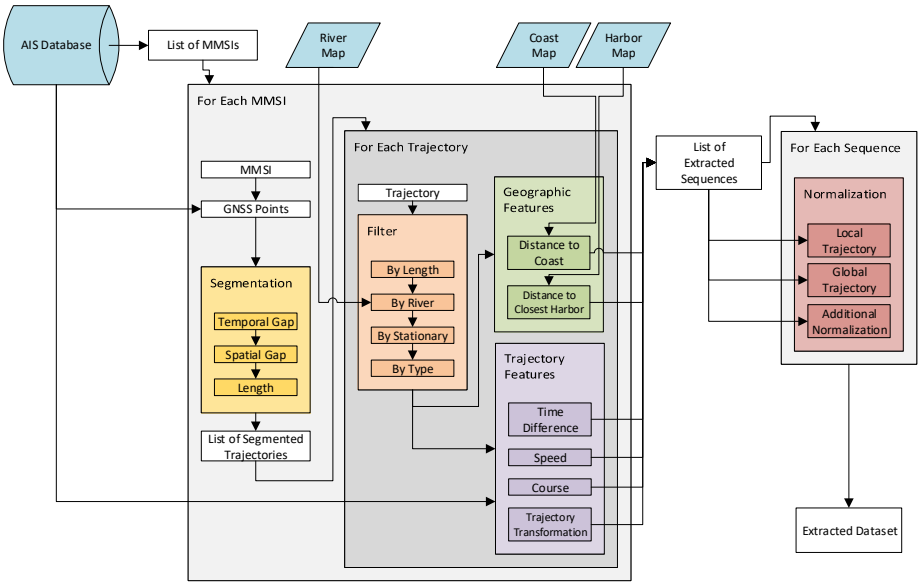



Figure 1: Visualization of all preprocessing steps. Input in blue, segmentation in yellow, filtering in orange, geographic features in green, trajectory feature in purple and normalization in red.

### 3.1 Trajectory features

Four different trajectory features are used:

- Time difference
- Speed over ground
- Course over ground
- Trajectory transformation

As the incoming data from AIS is not necessarily uniformly distributed in time, there is a need to create a feature representing the time dimension. Therefore, the time difference between two samples is introduced.

As the speed and course over ground is directly accessible through the AIS data, the network will be directly fed with these features. The vessel's speed is a numeric value in 0.1-knot resolution in the interval $[0; 1022]$ and the course is the negative angle in degrees relative to true north and therefore in the interval $[0; 359]$.

The position will be transformed in two ways. The first transformation, further called "relative-to-first", will shift the trajectory to start at the origin. The second transformation, henceforth called "rotate-to-zero", will rotate the trajectory, in such a way, that the end point is on the x-axis.

### 3.2 Geographic features

Additional to the trajectory based features, two geographic features are derived by using coastline maps[6] and a map of large harbours. The coastline map consists of a list of line strips. In order to reduce complexity, the edge points are used to calculate the "Distance-To-Coast". Further, only a lower resolution of the shapefile itself is used. In Figure 2, the resolution "high" and "low" for some fjords in Norway are shown. Due to the geoindex' cell size set to 40 km, a radius of 20 km can be queried.

Figure 2: Comparison between the "high" (left) and "low" (right) resolution of the coastline.

The world's 140 major harbours based on the world port index[7] are used to calculate the "Distance-to-Closest-Harbor". As fishing vessels are expected to stay near to a certain harbour, this feature should support the network to identify some shiptypes. The geoindex' cell size is set for this feature to 5,000 km, resulting in a maximum radius of 2,500 km.

### 3.3 Segmentation

The data is split into separate trajectories by using gaps in either time or space, or the sequence length. As real AIS data is used, package loss during the transmission is common. This problem is tackled by splitting the data

[6] http://www.soest.hawaii.edu/wessel/gshhg/
[7] https://msi.nga.mil/Publications/WPI

– if the time between two successive samples is larger than 2 hours, or
– if the distance between two successive samples is large.

Regarding the distance, even though the great circle distance is more accurate, the euclidean distance is used. For simplification the distance value is squared and as a threshold $10^{-4}$ is used. Depending on latitude this corresponds to a value of about 1 km at the equator and only about 600 m at 60° N. Since the calculation includes approximation a relatively high threshold is chosen.

As the neural network depends on a fixed input size, the data is split into fitting chunks by cutting and padding with these rules:

– Longer sequences are split into chunks according to the desired sequence length.
– Any left over sequence shorter than 80 % of the desired length is discarded.
– The others will be padded with zeroes.

This results in segmented trajectories of similar but not necessarily same duration.

### 3.4 Filter

As this work is about the vessel behaviour at sea, stationary vessels (anchored and moored vessels) and vessels traversing rivers are removed from the segmented trajectories. The stationary vessels are identified by using a measure of movement in a trajectory:

$$\alpha_{\text{stationary}} = \frac{\sum_{i=1}^{n} |P_i - P_{i-1}|}{n}, \tag{1}$$

where $n$ as the sequence length and $P_i$ its data points. A trajectory will be removed if $\alpha_{\text{stationary}}$ is below a certain threshold.

A shapefile[8] containing the major and most minor rivers (compare **??**) is used in order to remove the vessels not on the high seas. A sequence with more than 50 % of its points on a river is removed from the dataset.

### 3.5 Normalization

In order to speed up the training process, the data is normalized in the interval $[0; 1]$ by applying

$$X' = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}. \tag{2}$$

Here, for the positional features a differentiation between “global normalization” and “local normalization” is taken into account. The “global normalization” will scale the input data for the maximum $X_{\text{max}}$ and minimum $X_{\text{min}}$ calculated over the entire data set, while “local normalization” will estimate the maximum $X_{\text{max}}$ and minimum $X_{\text{min}}$ only over the trajectory itself. As the data is processed parallel, the parameters for the “global normalization” will be calculated only for each chunk of data. This will result in slight deviations in the minimum and maximum, but for large batches this should be neglectable.

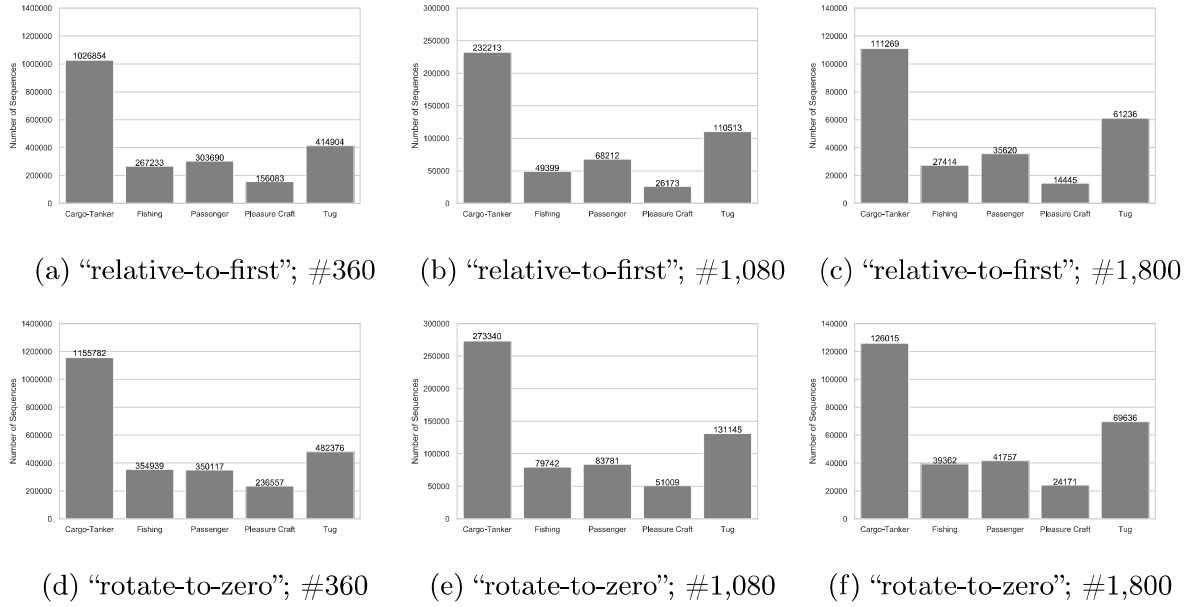
All other additional features are normalized as well. For the geographic features "Distance-to-Coast" and "Distance-to-Closest-Harbor" the maximum distance, that can be queried depending on grid size, is used as $X_{max}$ and 0 is used as the lower bound $X_{min}$.

The time difference feature is scaled using a minimum $X_{min}$ of 0 and the threshold for the temporal gap since this is the maximum value for this feature. Speed and course are normalized using 0 and their respective maximum values.

[8] http://www.soest.hawaii.edu/wessel/gshhg/

### 3.6 Resulting dataset

For the dataset, a period between 2018-07-24 and 2018-11-15 is used. Altogether 209,536 unique vessels with 2,144,317,101 raw data points are included. Using this foundation and the previously described methods, six datasets are derived. All datasets use the same spatial and temporal thresholds. In addition, filter thresholds are identical as well. The datasets differentiate in their sequence length and by applying only the "relative-to-first" transformation or additionally the "rotate-to-zero" transformation. Either 360, 1,080, or 1,800 points per sequence are used resulting in approximate 1 h, 3 h, or 5 h long sequences. In Figure 3, the distribution of shiptypes in the datasets after applying the different filters is shown.




(a) "relative-to-first"; #360 (b) "relative-to-first"; #1,080 (c) "relative-to-first"; #1,800

(d) "rotate-to-zero"; #360 (e) "rotate-to-zero"; #1,080 (f) "rotate-to-zero"; #1,800

Figure 3: Number of sequences per class.

## 4 Neural Network Design

For the shiptype classification, neural networks are chosen. The different networks are implemented using Keras [14] with TensorFlow as backend [15].

Fawaz et al. [16] have shown, that, despite their initial design for image data, a residual neural network (ResNet) can perform quite well on time-series classification. Thus, as foundation for the evaluated architectures the ResNet is used. The main difference to other neural network architectures is the inclusion of "skip connections". This allows for deeper networks by circumventing the vanishing gradient problem during the training phase.

Based on the main idea of a ResNet, several architectures are designed and evaluated for this work. Some information regarding the structure are given in Table 1. Further, the single architectures are depicted in Figures 4a to 4f.

The main idea behind these architectures is to analyse the impact of the depth of the networks. Furthermore, as the features itself are not necessarily logically linked with

each other, the hope is to be able to capture the behaviour better by splitting up the network path for each feature.

To verify the necessity of CNNs two multilayer perceptron (MLP) based networks are tested: One with two hidden layers and one with four hidden layers, all with 64 neurons and fully connected with their adjacent layers. The majority of the parameters for the two networks are bound in the first layer. They are necessary to map the large number of input neurons, e. g., for the 360 samples dataset $360 * 9 = 3{,}240$ input neurons, to the first hidden layer.

Table 1: Parameter of neural network architectures.

| Name | Depth | # Parameters |
|---|---|---|
| Tiny ResNet | 11 | 29,125 |
| Shallow ResNet | 21 | 440,837 |
| Deep ResNet | 66 | 1,327,877 |
| Stretched Deep ResNet | 66 | 3,280,645 |
| Split ResNet | 26 | 390,701 |
| Total Split ResNet | 26 | 364,461 |
| MLP for 360 samples | 4 | 211,909 |
| deeper MLP for 360 samples | 6 | 220,229 |

## 5 Training

Each of the datasets is split into three parts: 64 % for the training set, 16 % for the validation set, and 20 % for the test set. For solving or at least mitigating the problem of overfitting, regularization techniques (input noise, batch normalization, and early stopping) are used.

Small noise on the input in the training phase is used to support the generalization of the network. For each feature a normal distribution with a standard deviation of 0.01 and a mean of 0 is used as noise.

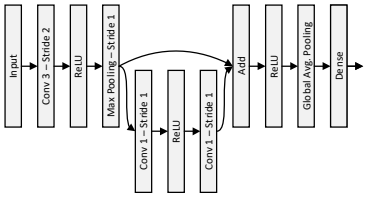
Furthermore, batch normalization is implemented. This means, before each ReLU-layer a batch normalization layer is added, allowing higher learning rates. Therefore, the initial learning rate is doubled. Additionally, the learning rate is halved if the validation error does not improve after ten training epochs, improving the training behaviour during oscillation on a plateau.

In order to prevent overfitting, an early stopping criteria is introduced. The Training will be interrupted if the validation error is not decreasing after 15 training epochs.

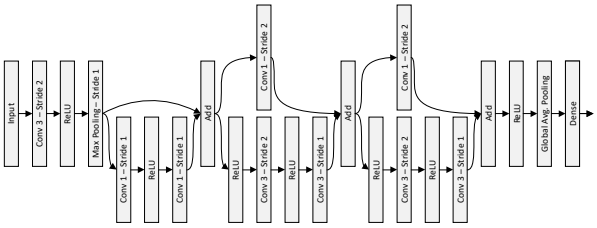
To counter the dataset imbalance, class weights were considered but ultimately did not lead to better classification results and were discarded.

## 6 Evaluation

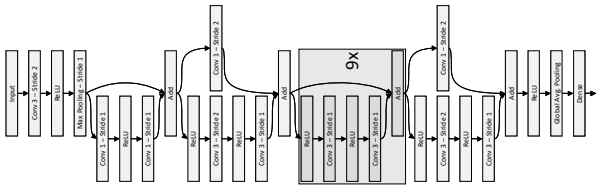
The different neural network architectures are evaluated on a AMD Ryzen Threadripper 1920X 12-Core Processor with 64 GB of memory and 4x Nvidia Geforce GTX 1080 Ti. Each network is evaluated on the six datasets. For the ResNet based networks, the



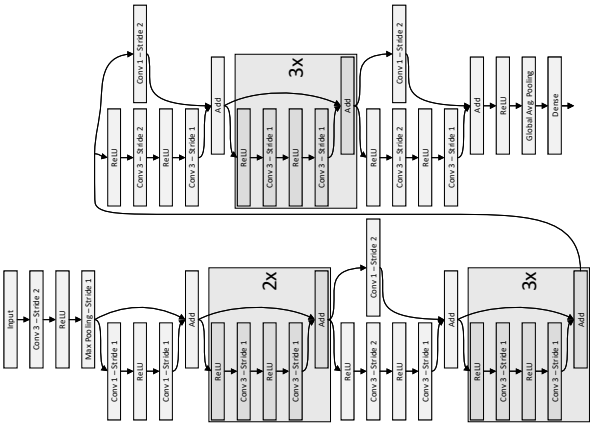

(a) Tiny ResNet



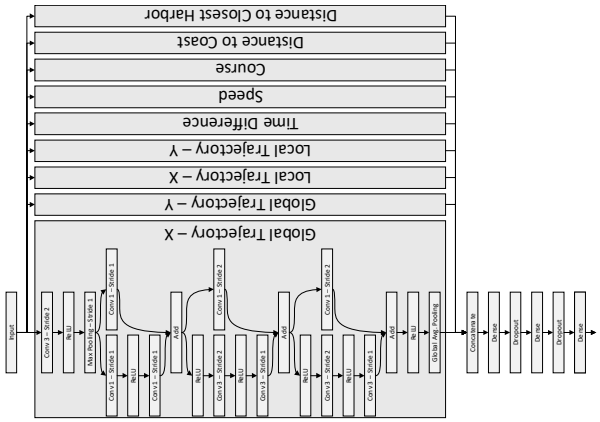

(b) Shallow ResNet



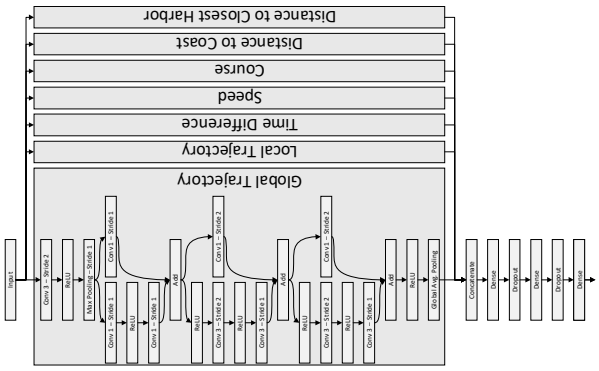

(c) Deep ResNet




(d) Stretched Deep ResNet




(e) Split ResNet

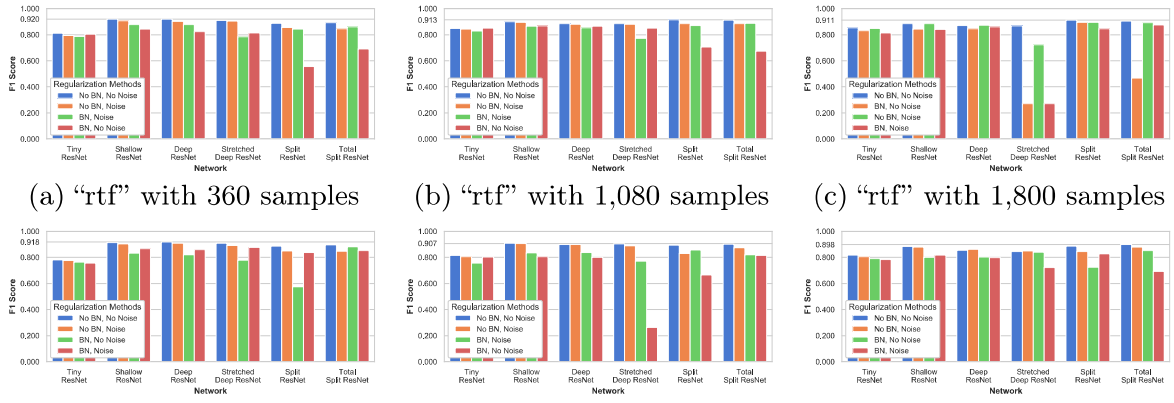



(f) Split ResNet

Figure 4: Schematic architectures.

batch normalization and the input noise is tested. The initial learning rate is set to 0.001 without batch normalization and 0.002 with batch normalization activated. The maximum number of epochs is set to 600. The batch sizes are set to 64, 128, and 256 for 360, 1,080, and 1,800 samples per sequence respectively.

In total 144 different setups are evaluated. Furthermore, 4 additional networks are trained on the 360 samples dataset with "relative-to-first" transformation. Two MLPs to verify the need of deep neural networks, and the Shallow and Deep ResNet trained without geographic features to measure the impact of these features.




(a) "rtf" with 360 samples (b) "rtf" with 1,080 samples (c) "rtf" with 1,800 samples

(d) "rtz" with 360 samples (e) "rtz" with 1,080 samples (f) "rtz" with 1,800 samples

Figure 5: $F_1$-Scores of all networks. In addition, the regularization methods used are shown. The first row shows the results for the "relative-to-first" (rtf) transformation, the second for the "rotate-to-zero" (rtz) transformation.

The results for the six different architectures are depicted in Figure 5. For 360 samples the Shallow ResNet and the Deep ResNet outperformed the other networks. In case of the "relative-to-first" transformation (see Figure 5a), the Shallow ResNet achieved an $F_1$-Score of 0.920, while the Deep ResNet achieved 0.919. For the "rotate-to-zero" transformation (see Figure 5d), the Deep ResNet achieved 0.918 and the Shallow ResNet 0.913. In all these cases the regularization methods lead to no improvements.

The "relative-to-first" transformation performs slightly better overall. For the datasets with 360 samples per sequence, the standard ResNet variants achieve higher $F_1$-Scores compared to the Split ResNet versions. But this difference is relatively small. As expected, the Tiny ResNet is not large and deep enough to classify the data on a similar level.

For the "relative-first" transformation and trajectories based on 1080 samples (see Figure 5b), the Split ResNet and the Total Split ResNet achieve the best results. The first performed well with an $F_1$-Score of 0.913, while the latter is slightly worse with 0.912. In both cases again the regularization did not improve the result. For the "rotate-to-zero" transformation (see Figure 5e), the Shallow ResNet achieved an $F_1$-Score of 0.907 without any regularization and 0.905 with only the the noise added to the input.

For the largest sequence length of 1,800 samples, the split based networks slightly outperform the standard ResNets. For the "relative-to-first" transformation (see Figure 5c), the Split ResNet achieved an $F_1$-Score of 0.911, while for the "rotate-to-zero" transformation (see Figure 5f) the Total Split ResNet reached an $F_1$-Score of 0.898. Again without noise and batch normalization.

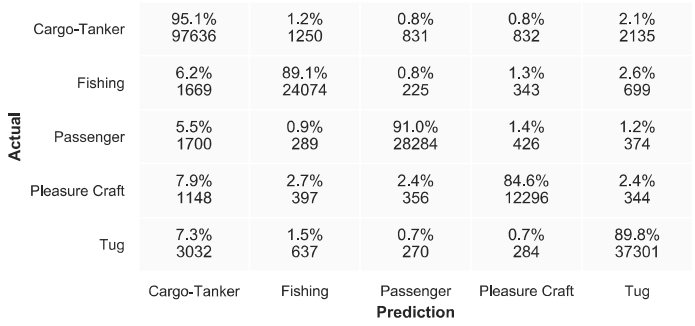



Figure 6: Confusion matrix of the Shallow ResNet on 360 samples with "relative-to-first" transformation, without added input noise and batch normalization.

To verify, that the implementation of CNNs is actually necessary, additional tests with MLPs were carried out. Two different MLPs are trained on the 360 samples dataset with "relative-to-first" transformation since this dataset leads to best results for the ResNet architectures. Both networks lead to no results as their output always is the "Cargo-Tanker" class regardless of the actual input. The only thing the models are able to learn is, that the "Cargo-Tanker" class is the most probable class based on the uneven distribution of classes.

An MLP is not the right model for this kind of data and performs badly. The large dimensionality of even the small sequence length makes the use of the fully connected networks impracticable. Probably, further hand-crafted feature extraction is needed to achieve better results.

To measure the impact the feature "Distance to Coast" and "Distance to Closest Harbor" have on the overall performance, a Shallow ResNet and a Deep ResNet are trained on the 360 sample length data set with the "relative-to-first" transformation excluding these features. The trained networks have $F_1$-Scores of 0.888 and 0.871 respectively. This means, by including this features, we are able to increase the performance by 3.5 %.

## 7 Discussion

The "relative-to-first" transformation compared to the "rotate-to-zero" transformation yields the better results. Especially, this is easily visible for the longest sequence length. A possible explanation can be seen in the "stationary" filter. This filter removes more trajectories for the "relative-to-first" transformation than for the additional "rotate-to-zero" transformation. A problem might be, that the end point is used for rotating the trajectory. This adds a certain randomness to the data, especially for round trip sequences.

In some cases, the Stretched Deep ResNet is not able to learn the classes. It is possible, that there is a problem with the structure of the network or the large number of parameters. Further, there seems to be a problem with the batch normalization, as seen in Figures 5c and 5e.

The overall worse performance of the "rotate-to-zero" transformation could be because of the difference in the "stationary" filter. In the "rotate-to-zero" dataset, fewer sequences are filtered out. The filter leads to more "Fishing" and "Pleasure Craft" sequences in relation to each other as described in section 3.6. This could also explain the difference in class prediction distribution since the network is punished more for mistakes in these classes because more classes are overall from this type.

For the evaluation, the expectation based on previous work by other authors was, that the shorter sequence length should perform worse compared to the longer ones. Instead the shorter sequences outperform the longer ones. The main advantages of the shorter sequences are essentially the larger number of sequences in the dataset. For example the 360 samples dataset with "relative-to-first" transformation contains about 2.2 million sequences, while the corresponding 1,800 sample dataset contains only approximately 250,000 sequences.

In addition, the more frequent segmentation can yield more easily classifiable sequences: The behaviour of a fishing vessel in general contains different characteristics, like travelling from the harbour to the fishing ground, the fishing itself, and the way back. The travelling parts are similar to other vessels and only the fishing part is unique. A more aggressive segmentation will yield more fishing sequences, that will be easier to classify regardless of observation length.

The Shallow ResNet has the overall best results by using the 360 samples dataset and the "relative-to-first" transformation. The results for this setup are shown in the confusion matrix in Figure 6. As expected, the Tiny ResNet is not able to compete with the others. The other standard ResNet architectures performed well, especially on shorter sequences.

The Split architectures are able to perform better on datasets with longer sequences, with the Shallow ResNet achieving similar performance. Comparing the number of parameters, all three architectures have about 400,000 the Shallow ResNet about 50,000 more, the Total Split ResNet about 40,000 less.

Only on the dataset with more sequences, the Deep ResNet performs well. This correlates with the need of more information due to the larger parameter count. Due to the reduced flexibility, the Split architecture can be interpreted as a "head start". This means, that the network has already information regarding the structure of the data, which in turn does not need to be extracted from the data. This can result in a better performance for smaller datasets.

All in all, the best results are always achieved by omitting the suggested regularization methods. Nevertheless, the batch normalization had an effect on the learning rate and needed training epochs: The learning rate is higher and less epochs are needed before convergence.

## 8 Conclusion

Based on the ResNet, several architectures are evaluated for the task of shiptype classification. From the initial dataset based on AIS data with over 2.2 billion datapoints six datasets with different trajectory length and preprocessing steps are derived. Further to the kinematic information included in the dataset, geographical features are generated.

Each network architecture is evaluated with each of the datasets with and without batch normalization and input noise. Overall the best result is an $F_1$-Score of 0.920 with the Shallow ResNet on the 360 samples per sequence dataset and a shift of the trajectories to the origin. Additionally, we are able to show, that the inclusion of geographic features yield an improvement in classification quality.

The achieved results are quite promising, but there is still some room for improvement. First of all, the the sequence length used for this work might still be too long for real world use cases. Therefore, shorter sequences should be tried. Additionally, interpolation for creating data with the same time delta between two samples or some kind of embedding or alignment layer might yield better results. As there are many sources

for additional domain related information, further research in the integration of these sources is necessary.

## Acknowledgments

Underlying projects to this article are funded by the WTD 81 of the German Federal Ministry of Defense. The authors are responsible for the content of this article.

This work was developed in the Fraunhofer Cluster of Excellence "Cognitive Internet Technologies".

## References

1. UNCTAD: Review of maritime transport 2019 (2019) last accessed 2019-11-19.
2. International Chamber of Shipping: Key Facts (2018) last accessed 2019-01-22.
3. Food and Agriculture Organization of the United Nations: Report of the second meeting of the regional working group on illegal, unreported and unregulated (IUU) fishing, Barbados, 19–21 September 2017 (2018) ast accessed 2019-01-22.
4. Harati-Mokhtari, A., Wall, A., Brooks, P., Wang, J.: Automatic Identification System (AIS): Data Reliability and Human Error Implications. Journal of Navigation **60**(3) (2007) 373–389
5. Riveiro, M., Pallotta, G., Vespe, M.: Maritime anomaly detection: A review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **8**(5) (2018)
6. Jiang, X., de Souza, E.N., Pesaranghader, A., Hu, B., Silver, D.L., Matwin, S.: Trajectorynet: An embedded gps trajectory representation for point-based classification using recurrent neural networks. In: Proceedings of the 27th Annual International Conference on Computer Science and Software Engineering, IBM Corp. (2017) 192–200
7. Jiang, X., de Souza, E.N., Liu, X., Soleimani, B.H., Wang, X., Silver, D.L., Matwin, S.: Partition-wise recurrent neural networks for point-based ais trajectory classification. Computational Intelligence **6** (2017)
8. Nguyen, D., Vadaine, R., Hajduch, G., Garello, R., Fablet, R.: A multi-task deep learning architecture for maritime surveillance using ais data streams. In: 2018 IEEE International Conference on Data Science and Advanced Analytics (DSAA). (2018)
9. Hu, B., Jiang, X., de Souza, E.N., Pelot, R., Matwin, S.: Identifying fishing activities from ais data with conditional random fields. In: Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on, IEEE (2016) 47–52
10. Saini, R., Roy, P.P., Dogra, D.P.: A segmental hmm based trajectory classification using genetic algorithm. Expert Systems with Applications **93** (2018) 169–181
11. Fischer, Y.: Wissensbasierte probabilistische Modellierung für die Situationsanalyse am Beispiel der maritimen Überwachung. PhD thesis, Karlsruhe Institute of Technology (2016)
12. Anneken, M., de Rosa, F., Kröker, A., Jousselme, A.L., Robert, S., Beyerer, J.: Detecting illegal diving and other suspicious activities in the North Sea: Tale of a successful trial. In: 2018 20th International Radar Symposium (IRS). (2019)
13. De Rosa, F., Ben Abdallah, N., Jousselme, A.L., Anneken, M.: Source quality handling in fusion systems: a Bayesian perspective. In: Maritime Big Data Workshop, NATO STO CMRE (May 2018)
14. Chollet, F., et al.: Keras. https://keras.io (2015)
15. Abadi, M., et al.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015)
16. Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.A.: Deep learning for time series classification: a review. Data Mining and Knowledge Discovery **33**(4) (2019) 917–963