Multi-Object Tracking with Machine Learning based trajectories prediction and soft track-association

Christian Alexander Holz, Christian Bader, Matthias Drüppel

***Abstract*—In this paper, we develop Machine Learning (ML)-based methods for Multi-Object Tracking (MOT) within the context of Advanced Driver Assistance Systems (ADAS). Given the increasing complexity and demand for precise and efficient object tracking systems in the automotive industry, this work focuses on the integration of ML techniques into established tracking methodologies. Key contributions encompass the creation and evaluation of three specialized neural networks: (i) the Prediction Network for predicting the trajectories of tracked objects, (ii) the Single Association Network (SANT) for associating single incoming sensor object to n existing tracks, (iii) and the Multi-Association Network (MANTa) for associating multiple sensor objects to the n existing tracks. We combine our ML methods with a traditional Kalman filter framework, offering a data driven approach to address MOT challenges while maintaining the modularity and interpretability of classical filter approaches. We asses both, the performance of all three models alone as well as their impact on the performance when they are integrated in the Kalman framework. The results reveal a modular, robust, and maintainable tracker, underscoring the potential of ML integration in ADAS Tracking Systems.**

(i) Our results for the Prediction Network show a reduction in the Root Mean Square Error on the KITTI tracking data set car by half compared to a basic Kalman Filter. (ii) The Single Association Network (SANT) was developed as a replacement of the Global Nearest Neighbor method and enables a purely data-based assignment method, achieving a validation accuracy of 95% on KITTI cars. (iii) The Multi-Association Network (MANTa) similarly achieves an accuracy of 95% on the same validation set, but enables an assignment of now m sensor objects to a set of n tracks.

***Index Terms*—Article submission, IEEE, IEEEtran, journal, LaTeX, paper, template, typesetting.**

I. Introduction

THE ongoing evolution of Advanced Driver Assistance Systems (ADAS) has brought the need for precise and reliable Multi Object Tracking (MOT) into the spotlight [1], [2]. In complex and dynamic environments, as encountered in urban traffic, it is crucial to simultaneously and accurately capture the positions and movements of multiple objects - a key challenge in computer vision (CV) for automated driving.

In the commonly used Tracking-by-Detection (TbD) paradigm, a tracker fuses detected sensor objects (SO) to create consistent object tracks over time. A crucial step within this paradigm is the association of the incoming measured SO with their corresponding existing object tracks to update their properties. If no association can be made, new object tracks must be initialized.

C. Holz is with Daimler Truck AG, Research and Advanced Development, Stuttgart, Germany

C. Bader is with Daimler Truck AG, Research and Advanced Development, Stuttgart, Germany

M. Drüppel is with the Center for Artificial Intelligence, Duale Hochschule Baden-Württemberg (DHBW), Stuttgart, Germany

Tracking frameworks form the heart of ADAS systems that are used in millions of vehicles around the globe. The vast majority of these frameworks rely on classical approaches such as the Kalman filter (KF) or its variants. These classical tracking theories have the great benefit of being modular and interpretable. The task is split into clearly separated subtasks such as the prediction of currently tracked objects and the association with newly measured ones. Furthermore, the math and the theory itself is often clean and comprehensible. However, in the automotive industry tracking systems are not developed for a single model, but are usually used as a platform and are deployed to a variety of different car models with different sensor sets, different installation heights of the sensors, used in different countries and must always perform for a wide range of driving scenarios. This usually leads to poor performance in certain scenes for a specific system. With classical systems that not learn directly from data, these situations are often solved by the implementation of heuristics and by tweaking parameters of the tracking system for the troublesome scenes. But from a software engineering point of view hand-engineered parameters and heuristics are extremely hard to maintain and develop further for new scenarios and new system configurations. Moreover, hand-engineered classical approaches can reach its limits for the overall performance and in challenging situations that would require a more automated and reproducible development strategy.

In this work, we propose a data driven approach to tracking frameworks, which would allow the same system to be fine tuned for specific configurations relying only on data, thus increasing maintainability and adaptability. We do this preserving one of the biggest strengths of classical approaches: its modularity, by replacing only single tracking components with ML models.

In contrast to the Kalman filter, our Prediction Network is capable of predicting the state of individual objects without the need for a predefined state or prediction model at runtime. The self-learning, data driven approach enables adaptability to various scenarios and the ability to effectively handle non-linearities. Many conventional tracking systems rely on static methods for data association. Commonly used algorithms like the Hungarian algorithm require heuristics and fixed thresholds. Our Single Association Network (SANT) replaces the calculation of a distance metric for the corresponding assignment by employing machine learning. Both, the Prediction Network and SANT can be developed and evaluated as stand-alone models. For a throughout evaluation, we proceed by integrating them into an existing tracking system and demonstrate their performance through multiple tests and comparisons with established methods.

This work provides new insights and advancement in the development of Advanced Driver Assistance Systems (ADAS), contributing to the further evolution of technologies for autonomous driving (AD).

II. Related work

A. Tracked object prediction

One fundamental problem in Tracking-by-Detection (TbD) frameworks is the prediction of the states of the already tracked objects. In many approaches Kalman filters and their variants have proven to be effective for state prediction [1]–[3]. However, they reach their limits in more complex scenarios, particularly in the presence of non-linear motion patterns and interactions among multiple objects. In this work, we introduce a novel MOT approach that leverages Machine Learning to overcome these challenges. We specifically focus on the development and implementation of Neural Networks (NN), which can enable more precise and flexible data-driven object tracking.

B. Association

Another fundamental problem for a TbD tracker is the data association. For this, some approaches only consider the current state of the tracks, while others integrate temporal information such as the track history. This aggregate of temporal information can be done for example using attention mechanisms as used in [4], [5] or recurrent neural networks (RNNs) as developed in [6], [7]. The latter is what we are also pursuing in this work. Similar to the problem statement in Mertz et al [7], the aim of our work is to develop a data-based approach that can learn to completely solve the combinatorial non deterministic polynomial time (NP) hard optimization problem of data association. Mertz et al [7] use a distance matrix based on the Euclidean distance measure as input for the developed association network, thus replacing an association algorithm such as the Hungarian Algorithm (HA) [8]. In the context of this work, we put forward the hypothesis that a Gated Recurrent Unit (GRU)-based association network can be designed and trained using an undefined distance measure.

C. Real time applications

ADAS are embedded real-time applications where it is crucial to predict the state of objects immediately after their detection. This rules out offline tracking methods as presented in [10] that process the entire video material at once in a batch process. Therefore, most recent approaches for tracking multiple objects rely on online methods that do not depend on future image information. Online methods use various features to estimate the similarity between the recognized objects and the existing tracks. This can be done on the basis of their predicted positions or even similarities in appearance.

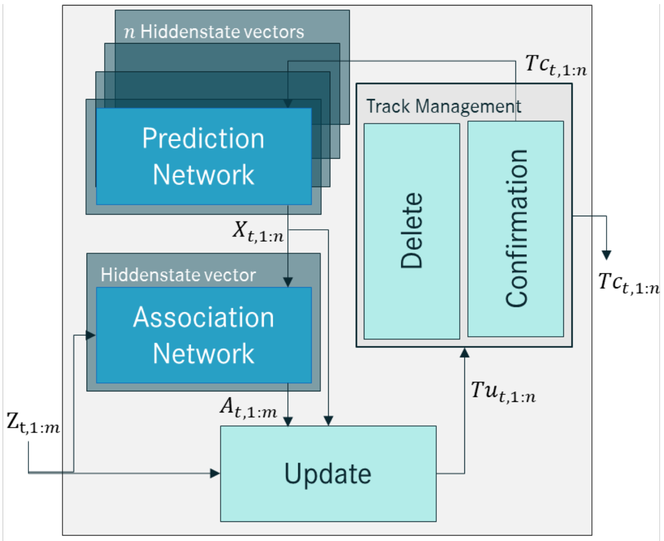
a) Kalman Filter based tracker: [1]–[3] propose a Kalman Filter based TbD multi-object tracker. [1] presents a MOT approach based on simple visual cues. The authors contend that many existing multi-object trackers are too complex and require a large amount of computational resources. Instead, they propose a simpler approach based on basic visual features such as color, shape and motion. These visual cues are used to track objects at the image level and make associations between frames.

b) Recurrent Neural Network (RNN) based Tracker: Similar to our methods [6], [7], [11] present approaches for online multi-target tracking using recurrent neural networks (RNNs). The approach presented in my Mertz et al. [7] focusses on the task of data association in a TbD framework. The developed DeepDA model represents an LSTM-based Deep Data Association Network to learn and perform the association of objects between frames. By learning association patterns from the data, the tracker can achieve robust and reliable tracking results even in highly disturbed environments. Mertz et al. use a distance matrix based on the Euclidean distance measure as input data for the developed DeepDA network, thus replacing an association algorithm such as the Hungarian Algorithm (HA). It can be assumed that the Euclidean distance measure was also used as the basis for generating the ground truth (GT) training data (distance matrices) and for the evaluation. However, this is not explicitly stated. It can therefore be argued that this preprocessing step deprives the network opportunity to follow a different association logic or to learn it data based.

c) Attention Mechanism based Tracker: Each of the papers [4], [5], [12]–[14] present approaches for a tracker that utilize the attention mechanism [15], for example to compute soft data association [4]. The main research focus of the paper [4] is on soft data association, which enables the tracker to make probabilistic associations between objects and account for uncertainties in the associations. Soft data association in the SoDA model works by using attention mechanisms to aggregate information from all detections in a given temporal window. This allows the model to learn long-term and highly interactive relationships between detections and tracks from large datasets without using complex heuristics and hyperparameters.

III. Overview of our proposed models

Our primary contribution is the development and evaluation of three NN that we label: (i) the Prediction Network, (ii) the Single Association Network (SANT), and (iii) the Multi-Association Network (MANTa). Figure 1 provides an overview of our approach in which the association network can be implemented using either SANT or MANTa. The input for the proposed Prediction Network (i) are the sensor objects in every time step. If new objects are detected, these are stored in a fixed-dimensional state vector that contains information such as object position, orientation and dimension. The Prediction Network predicts all vectors to the next time step where they are used as input to either the SANT or MANTa association networks. These provide the association matrix, that is used to update the tracks using the corresponding sensor objects or create new tracked objects. The Track Management can then decide to delete tracks, that were not updated for a specific amount of time and send out tracks to the next higher software component that have been confirmed by sensor objects.



$Z_{t,1:m}$	– *Sensorobjects* $1{:}m$ *at timestep* t
$X_{t,1:n}$	– *Network Predictions* $1{:}n$ *at timestep* t
$A_{t,1:m}$	– *Assignment of sensor object 1:m to Network Predictions* $1{:}n$ *at timestep* t
$Tu_{t,1:n}$	– *Updated Tracks* $1{:}n$ *at timestep* t
$Tc_{t,1:n}$	– *Confirmed Tracks* $1{:}n$ *at timestep* t

n Hiddenstate vectors for n confirmed tracks

Fig. 1. Schematic representation of the two integrated networks (blue) in a tracking-by-detection framework. The Association Network can either be filled with the our SANT (single) or MANTa (multi) association models.

IV. Tracking with Prediction and Association Networks

We apply the tracking-by-detection (TbD) paradigm, in which a tracker fuses object detections (sensor objects) to generate object tracks that are consistent over time. [3], for example, provides a framework for analyzing tracking approaches that follow the TbD paradigm. This was used accordingly in this study.

To enable our models to incorporate temporal information, we use Long Short-Term Memory (LSTM) and bidirectional Long Short-Term Memory (BiLSTM) networks. Both for the prediction and the association (SANT) of the sensor objects to the existing tracks.

A. Prediction Network

Our approach is to use the hidden states of the LSTM layer as an object-specific information storage. The Prediction Network was designed for an open-loop application, this means that the network predicts values for a future time step based on previously received data. For use in an online MOT process, the network is therefore able to make a prediction about the most likely next state values using the state values received for a given state vector of a sensor object.

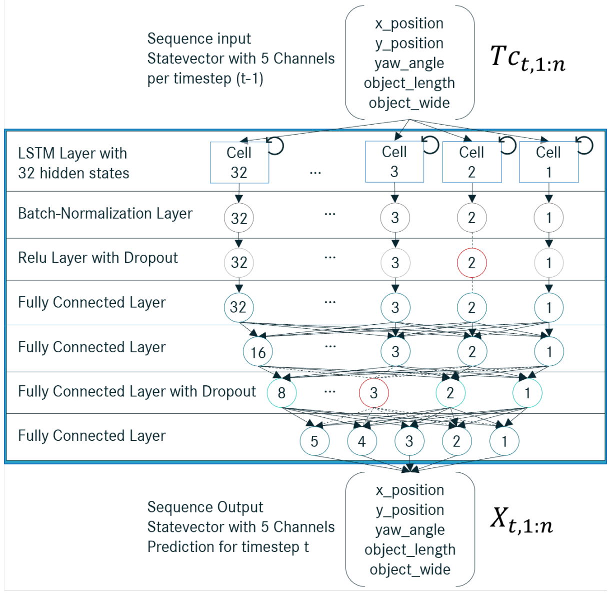


Fig. 2. Schematic representation of the generic prediction network structure.

a) Network architecture: The first layer of our Prediction Network is formed by the LSTM layer. The internal values (hidden states) of this layer are updated per time step based on the received measurement data. This updating process therefore provides a correction over the course of the sequence of each track. The number of hidden units (cells) corresponds to the amount of information that the layer remembers between time steps, as shown in Figure 2. The hidden states can contain information from all the previous time steps, regardless of the sequence length. The values of the hidden states of the LSTM layer are updated in each time step based on the received measurement data and the hidden states of the past time step. These updates therefore results in an internal correction over the course of the sequence. In our prediction network, we have placed a batch normalization layer directly after the LSTM layer. This means that the output of the LSTM layer is normalized before being passed on to the subsequent Relu layer. This helped speed up training and improve convergence by reducing internal covariate shifts [16]. A Relu layer performs a threshold operation to each element of the input, where any value less than zero is set to zero. At training time, a dropout layer randomly sets input elements to zero with a given probability providing a regularizing effect [17]. The Fully Connected (FC) Layer combines all of the local information learned by the previous layers. The last FC layer must be equal to the number of response variables in the output layer [18]. The basis for our loss is the mean-squared-error (MSE) calculated from the prediction for each state value. The MSE indicates the average of the squared difference between the model prediction and the target value, which we use as the measure of the quality of the prediction. For a single observation, the MSE is given by:

$$MSE = \frac{1}{R}\sum_{i=1}^{R}(y_i - \hat{y}_i)^2, \tag{1}$$

where R is the size of the predicted state vector, y_i is the ground truth value (KITTI car tracks) and $\hat{y}_i$ is the prediction of the network for training set sample i. The loss is evaluated for several sequences, each with numerous time stamps. For our Prediction Network the loss function is half the mean-square-error of the predictions added up for each time step in the training set, normalized by the total number of all time stamps in the used sequences S:

$$loss = \frac{1}{2S} \sum_{j=1}^{S} \sum_{i=1}^{R} (y_{ij} - \hat{y}_{ij})^2. \quad (2)$$

During training, the average loss is calculated using the observations in the mini-batch, so S equals the mini-batch length. On the KITTI data set (cars and vans), a RMSE of 0.025 was achieved for the position prediction of all objects in the dataset. Using a standard Kalman filter (KF), an RMSE of 0.066 was achieved on the same data set.

b) Data preprocessing: Ground truth (GT) data in the form of tracks of vehicles from the KITTI data set was used for our development. A GT Track contains the information of an object over time, starting with its appearance and ending with its exit from the sensor detection range. In order to achieve better generalization and to increase the chance that the training converges, the track state values at time t (predictors) and track state values at time $t+1$ (targets) were normalized following [19], such that the possible predictions and targets have a mean value of zero and a unit variance. The mean value μ and the standard deviation σ for each state variable were calculated for all tracks using the following equations:

$$\mu = \frac{1}{R} \sum_{i=1}^{R} S_i \quad \text{and} \quad \sigma = \sqrt{\frac{1}{R-1} \sum_{i=1}^{R} |S_i - \mu|^2} \quad (3)$$

where S_i the total number of all time stamps of all tracks and R is the number of states per track.

We also apply pre-padding as described in [20] where Reddy et al. show how padding influences the performance of neural networks in sequence-based tasks. Their study suggests that although both pre-padding and post-padding are feasible, the choice of padding technique can have a significant impact on the efficiency of the model, especially for LSTM networks where the sequence context is crucial.

To handle sequences of varying lengths in our LSTM network, we utilized the padding technique. In this process, shorter sequences are padded with special padding tokens so that all sequences within a batch have the same length. We used zeros as padding tokens, which were appended to the end of a sequence as needed. This allows for efficient batch processing and ensures consistent training of the network without the model performance being affected by the varying sequence lengths. This allows for efficient batch processing and ensures consistent training of the network without the model performance being affected by the varying sequence lengths.

As [20] described, padding can lead to noise in the network, but a sequence adjustment per mini-batch is required for training LSTM networks. To minimize the effect, the sequences of the training data were therefore sorted by length before the mini-batch padding. Figure 3 clearly shows that the padding (turquoise area) is minimized by a sorted training data set per mini-batch. Without this modification, no convergence could be achieved in the training.

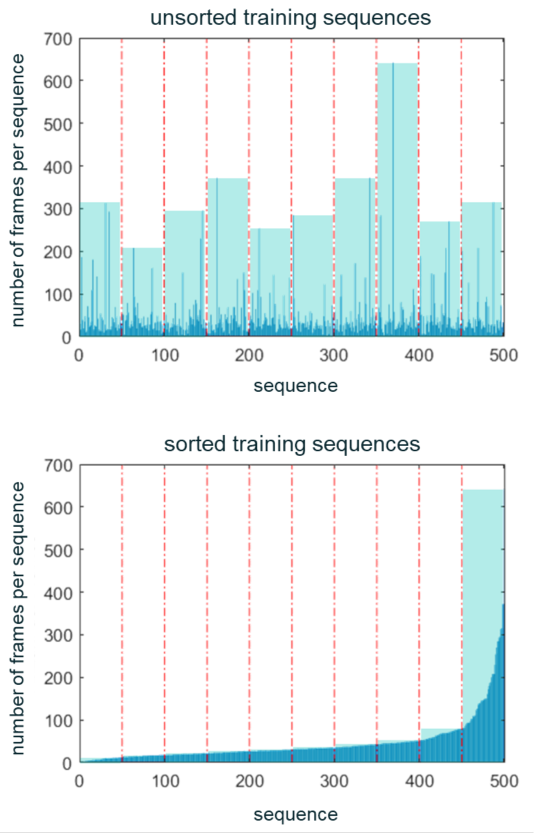


Fig. 3. The impact of padding on unsorted and sorted training data. The figure illustrates the padding effect in an LSTM network when sequences have varying lengths. Top: Unsorted data results in a high padding requirement, as shorter sequences need substantial padding to match the length of the longest sequence in the batch. Bottom: Sorted data minimizes the padding requirement by grouping sequences of similar length within the same batch. This leads to more efficient use of computational resources and can improve training efficiency and model performance.

B. Single Association Network (SANT)

The association network enables a data-based approach to solve the combinatorial non deterministic polynomial time (NP) hard optimization problem of data association. Compared to other association methods such as [6], [7] our SANT model is an approach that does not receive a defined distance matrix as input data, but instead receives the existing tracks and each new measured sensor objects.

This eliminates the need to define a distance measure, giving the network the freedom to follow its own association logic and learn it based on the training data. Therefore, the association network replaces the calculation of a distance metric and an association algorithm such as the Hungarian Algorithm (HA) [8].

a) Data preprocessing: In our development of SANT, the data association was considered as a time-structured problem (sequence-to-vector) where exactly one sensor object $Z_{(t,1)}$ is associated to a set of tracks $X_{(t,1:n)}$. The corresponding tracks were extracted from the KITTI data set of the labelled camera recordings. However, no real GT data exist for the association problem under consideration. To ensure that the assignment of a sensor object to one track of a given set of tracks has exactly one correct solution, the senor object was generated in a time step from the existing track set of a time step. The data was noised artificially in order to generate realistic sensor data from the GT data. All existing GT tracks were noised one by one for each time step. To achieve this, a maximum of 3% of the value of the current state vector was randomly subtracted or added to each value. The data set was created in 7 iterations, so that the noise intensity was increased by 0.5% per iteration.

As shown in figure 4 the data format was created accordingly to enable index-based track assignment for SANT. The size of the input matrix therefore corresponds to $m \times n + 1$. With $m = 5$ as the number of state values for our work and $n = 16$ as the maximum number of tracks per time step.

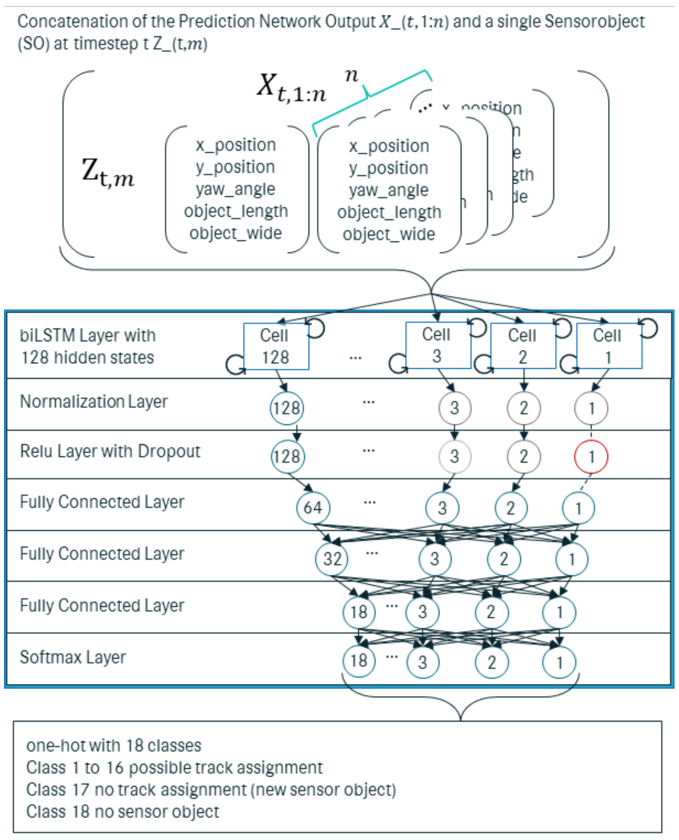

Concatenation of the Prediction Network Output X_(t,1:n) and a single Sensorobject (SO) at timestep t Z_(t,m)
$X_{t,1:n}$
n
$Z_{t,m}$
x_position
y_position
yaw_angle
object_length
object_wide
biLSTM Layer with 128 hidden states
Cell 128
Cell 3
Cell 2
Cell 1
Normalization Layer
Relu Layer with Dropout
Fully Connected Layer
Fully Connected Layer
Fully Connected Layer
Softmax Layer
128
64
32
18
3
2
1
one-hot with 18 classes
Class 1 to 16 possible track assignment
Class 17 no track assignment (new sensor object)
Class 18 no sensor object

Fig. 4. Schematic representation of the generic network structure of the Single Association Network (SANT).

b) Network architecture: The network as depicted in fig. 4 is designed as a sequence-to-classification network. At each time step, a matrix is passed as input holding both, the information of to-be-assigned sensor object and the currently tracked objects.

By combining the outputs of two LSTM layers that pass the information in opposite directions, [9] demonstrates the ability to capture the context from both ends of the sequence. The resulting architecture is called Bidirectional / Bidirectional Long Short-Term Memory Network (biLSTM). As part of these investigations, the best association performance was achieved with the biLSTM layer. Architectures with RNN, GNU, LSTM and biLSTM layers were tested. The network shown in Figure 4 achieved the best performance in comparison with a Training and Validation Accuracy of 95%. The output mode 'last' has been configured in the biLSTM layer. This layer is therefore able to receive a sequence as input and output a single value or value vector. This form of dimension reduction is necessary in order to carry out a corresponding classification.

The last fully connected layer (FC) specifies the number of classes via the number of output values. The classes are calculated in the softmax layer by applying the softmax function resulting in a probability distribution. The softmax function converts a number of values z_i into a probability vector with i values.

By using the cross-entropy operation in the output layer, SANT could be trained to a correct assignment value (GT class index). The cross-entropy cost function (cross-entropy) calculates the cross-entropy loss between network predictions and target values for the unique assignment task (for mutually exclusive classes): One-hot coding is used to represent the class in binary form in a vector and thus generate a 1-to-n code. The following formula is used to calculate the cross-entropy loss values for each input value Y_j and associated target value T_j element by element:

$$loss_j = -(T_j ln Y_j + (1 - T_j) ln(1 - Y_j)) \tag{4}$$

To obtain a scalar $loss$, all loss values $loss_j$ are totalled and divided by the number of samples N. Optionally, the loss values of defined samples can be weighted with a weighting factor w_j:

$$loss = \frac{1}{N} \sum loss_j w_j \tag{5}$$

A respective use of the weighting factor w_j can be helpful for data sets with an imbalanced class distribution. In this study, no significant improvements could be achieved by adjusting the weighting factor.

C. Association network for multiple sensor objects

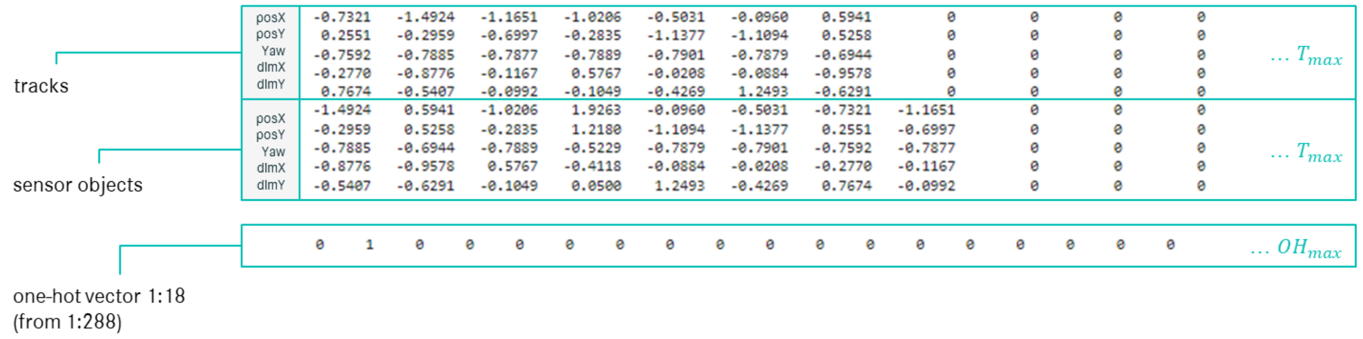
The developed single association network (SANT) shows that a data-based association logic can be learnt from a deep learning model. The aim was also to develop a network to recognise a number of m sensor objects SO to an existing number of n tracks in one operation step. in one operation step. A multi-association network (MANTa) was developed, which is able to solve the following association problems:

- **1 to n** - one SO to n tracks
- **m to 1** - m SO to one track
- **m to n** - m SO to n tracks
- **m to 0** - m SO to no tracks
- **0 to n** - no SO to n tracks

- **0 to 0** - no SO to no tracks

With the integration of a multi-association network (MANTa) into a Multi-Object Tracking (MOT) framework, the question can be asked whether a 0 to n and 0 to 0 assignment is a task to be solved. If no new SO is detected, the prediction of the last operation step can be continued until the track is deleted on the basis of the decreasing probability of existence within the track management module. The association algorithm or the MANTa does not need to be called if no tracks and sensor objects are detected. Although these assignment options can therefore be resolved via the programme structure in the MOT framework, these options are also taken into account. This is intended to ensure that the network also learns to deal with SOs and tracks that are no longer available.

a) Data preprocessing: The data set for the training and validation of MANTa was created according to the described objective. Figure 5 shows the input data structure with corresponding association tasks in the displayed time step (85) of sequence 20 of the KITTI data set. The respective tracks per time step were extracted across all sequences. The extracted tracks were each modified with noise as in the SANT development and then normalised.


tracks
sensor objects
one-hot vector 1:18
(from 1:288)

posX	-0.7321	-1.4924	-1.1651	-1.0206	-0.5031	-0.0960	0.5941	0	0	0	0	
posY	0.2551	-0.2959	-0.6997	-0.2835	-1.1377	-1.1094	0.5258	0	0	0	0	
Yaw	-0.7592	-0.7885	-0.7877	-0.7889	-0.7901	-0.7879	-0.6944	0	0	0	0	… T_{max}
dlmX	-0.2770	-0.8776	-0.1167	0.5767	-0.0208	-0.0884	-0.9578	0	0	0	0	
dlmY	0.7674	-0.5407	-0.0992	-0.1049	-0.4269	1.2493	-0.6291	0	0	0	0	
posX	-1.4924	0.5941	-1.0206	1.9263	-0.0960	-0.5031	-0.7321	-1.1651	0	0	0	
posY	-0.2959	0.5258	-0.2835	1.2180	-1.1094	-1.1377	0.2551	-0.6997	0	0	0	
Yaw	-0.7885	-0.6944	-0.7889	-0.5229	-0.7879	-0.7901	-0.7592	-0.7877	0	0	0	… T_{max}
dlmX	-0.8776	-0.9578	0.5767	-0.4118	-0.0884	-0.0208	-0.2770	-0.1167	0	0	0	
dlmY	-0.5407	-0.6291	-0.1049	0.0500	1.2493	-0.4269	0.7674	-0.0992	0	0	0	

0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 … OH_{max}

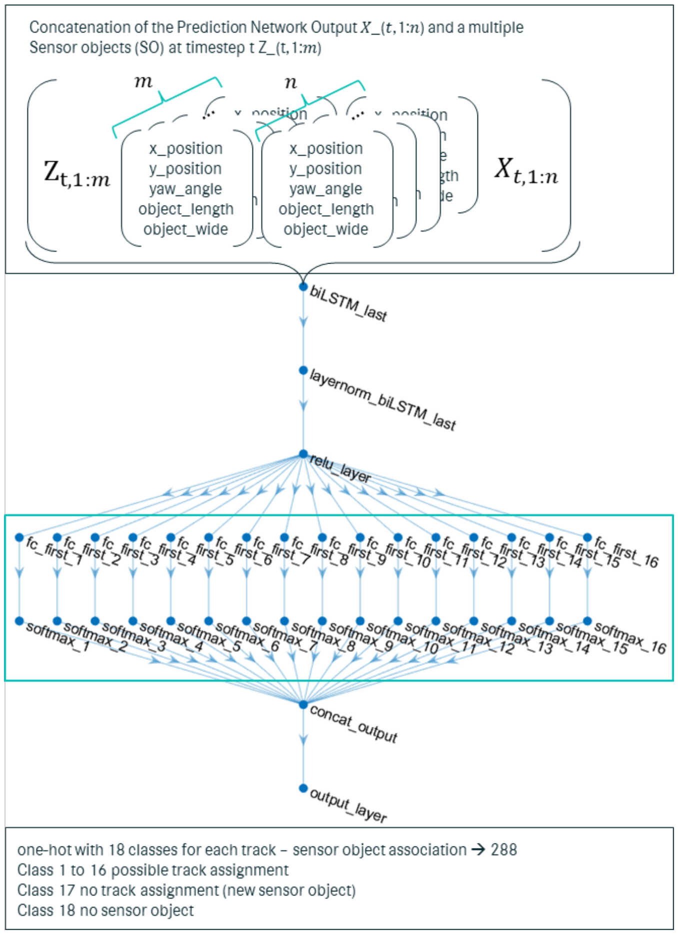
Fig. 5. MANTa, data structure, shows the non-noisy sensor objects to enable a visual assignment and increase understanding of the association procedure.

Figure 5 shows the non-noisy sensor objects to enable a visual assignment and increase understanding of the association procedure. The values obtained in this way were assigned as shown in Fig. 5 as measurements in pseudo-random order per time step assigned to an $F * T_{max}$ matrix. Where F represents the number of features. The number of features results from the status values per track and SO set. (here, $F = 2 * m$, with $m = 5$ (number of state values in this investigation)). T_{max} stands for the maximum number of existing tracks per time step. For the defined test case with cars and vans, the KITTI data set results in $T_{max} = 16$. There are 7 tracks in the time step of the sequence shown. Each track receives a new measurement in this time step, and an additional object was detected (new sensor object).

The GT assignment is displayed at the bottom of the section. The size of $OH_max = 288$ results from the maximum number of tracks $T_max = 16$ and the number of possible assignment $classes = 18$. The assignment classes result from the described index class 1 to 16 and additional degrees of freedom. One degree of freedom of the assignment represents the case that no measurement exists, another that the measurement should not be assigned. The section of the one-hot vector shown (1:18) thus shows the GT assignment of the first sensor object to the track at position two.

b) Network development: The schematic representation 6 shows the developed network architectures for the simultaneous association of a large number of sensor objects to a large number of tracks. This is what we call a Multi-Association Network (MANTa).

The BILSTM layer processes the input data as already explained for SANT. The task of associating a sensor object



one-hot with 18 classes for each track – sensor object association → 288
Class 1 to 16 possible track assignment
Class 17 no track assignment (new sensor object)
Class 18 no sensor object

Fig. 6. Schematic representation of the generic network structure of MANTa.

list with a track list requires a separate network part for each track. This extension is labelled accordingly in the graphic 6. For each track (from 1 to T_{max}), the MANTa has been developed with the familiar Fully Connected (FC), Softmax stack. Each softmax output consists of a vector with $classes = 18$ elements, which represents the most probable assignment. This means that a single assignment can be realised for each track. The vectors 1 to T_{max} are linked together in the Concatenation Layer. This creates a vector with 288 elements, whereby 18 elements each represent the most probable assignment of a measurement to a track.

The cost function was implemented according to the format of the GT and output data of the network. The cross-entropy loss function already introduced was used and a clear assignment was realised by means of additional iteration per time step through the respective track blocks. The following formula is used to calculate the cross-entropy loss values for each input value Y_i and associated target value T_i element by element. To obtain a scalar per time step $loss_k$, all loss values are summed up and divided by the number of $classes$. This results in the average loss per time step for all tracks.

$$loss_k = \frac{1}{classes} \sum_{i=1}^{classes} -(T_i ln Y_i + (1 - T_i) ln(1 - Y_i)) \quad (6)$$

Then, all scalars obtained per time step are summarised and divided by the number of samples N of a minibatch:

$$loss = \frac{1}{N} \sum loss_k \quad (7)$$

With the described procedure, the Multi-Association Network (MANTa) could be trained, for assigning a list of sensor objects SO to a list of tracks in one operation step.

V. Experimental Evaluation

In relation to the entire KITTI data set (Cars and Vans objects), MANTa achieves an average allocation accuracy of 70%. The main reason for this limitation was identified by analysing the extracted data. Figure 7 shows a distribution of the number of existing tracks per time step. For example, it can be seen that time steps containing a track account for almost a third of the entire available data set with 29.9% and an absolute number of 2315 samples.

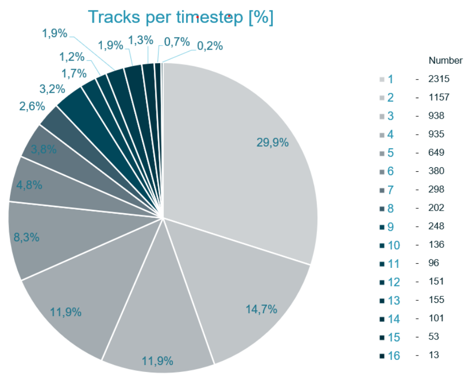


Fig. 7. KITTI Cars and Vans diagramm, distribution of available data, number of tracks per sample

Time steps containing one to six tracks together make up 81.5% of the samples. Accordingly, tests were carried out with a reduced data set in order to demonstrate the multi-association capability of the network. MANTa correctly assigns 95% of the data set with time steps containing one to six tracks. This confirms MANTa's ability to assign data with the appropriate data set, because SANT also achieves a validation accuracy of approx. 95% in relation to the entire KITTI data set (cars and vans).

VI. Conclusion

The developed networks can be modularly integrated into the tracking-by-detection (TbD) framework and thus replace classic heuristic algorithms within the MOT process.

SPENT replaces the state predictions of the Kalman filter KF. The trained network estimates the predictions per time step without the need for a dynamics model. The implementation allows the recurrent network to update the internal hidden states per time step, thus achieving an accurate state prediction without an external correction.

The model is suitable for use in real time applications and represents an alternative approach to classical prediction methods. The network verification shows an RMSE of 0.026 averaged over the training, validation and test data set. In relation to the predictions of the positions of an object in the X coordinate, this corresponds to an average deviation of 0.42m. In relation to the position in the Y direction relative to the ego vehicle, the average deviation of the verification performed is 0.23m.

For another main task of the TbD MOT method, data association, SANT was developed as a replacement for the classic GNN method. This means that SANT can replace the algorithms for calculating a distance metric and assignment procedures such as HA with the learned, training data-based assignment logic. Based on a defined validation data set with approx. 2700 samples, SANT achieves an accuracy of 95%.

MANTa is a further development of SANT and addresses the limitation of individual assignment. A network extension could be implemented that assigns a set of sensor objects m to a set of tracks n. The data situation was analysed in more detail and identified as a limitation for the network performance. The verification carried out shows that MANTa achieves an assignment accuracy of 95% in relation to the six most frequently occurring association sets.

Acknowledgments

We would like to express our sincere thanks to the organisations that provided financial support for this research project. In particular, we would like to thank Daimler Truck AG and the DHBW Stuttgart. Without this support, our work would not have been possible.

Our special thanks also go to our academic supervisors and mentors. I would like to thank Prof Dr Matthias Drüppel for his tireless support, valuable advice and continuous guidance throughout the duration of this project. His expertise and commitment have contributed significantly to the success of our research.

Further thanks go to my colleague. I would like to thank Christian Bader for his essential help in conducting the experiments and providing technical resources. His support and fruitful discussions have greatly enriched this work.

Finally, we would like to thank our families and friends for their unconditional support and encouragement. Their patience, understanding and motivation have seen us through the challenges of this project and have been an invaluable help.

APPENDIX
PROOF OF THE ZONKLAR EQUATIONS

PROOF OF THE FIRST ZONKLAR EQUATION

Appendix goes here.

PROOF OF THE SECOND ZONKLAR EQUATION

And here.

REFERENCES

[1] Jenny Seidenschwarz, Guillem Brasó, Victor Serrano, Ismail Elezi, Laura Leal-Taixé, "Simple cues lead to a strong multi-object tracker," *Paper*, 2022.
[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, Ben Upcroft, "Simple online and realtime tracking," *Paper*, 2016.
[3] Ba-Tuong Vo, "code set for research use: Multi-sensor multi-target tracking," 2013. [Online]. Available: https://ba-tuong.vo-au.com/codes.html
[4] W.-C. H. H. K. T.-Y. L. Y. C. R. Yu, "Soda: Multi-object tracking with soft data association," *Paper*, 2020.
[5] Q. C. W. O. H. L. X. W. B. L. N. Yu, "Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism," *Paper*, 2017.
[6] Anton Milan, Seyed Rezatofighi, Anthony Dick, Ian Reid, Konrad Schindler, "Online multi-target tracking using recurrent neural networks," *Paper*, 2016.
[7] H. L. H. Z. C. Mertz, "Deepda: Lstm-based deep data association network for multi-targets tracking in clutter," *Paper*, 2019.
[8] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, 1955.
[9] S. H. J. Schmidhuber, "Long short-term memory," *Neural Computation*, 1997.
[10] Roberto Henschel, Laura Leal-Taixé, Daniel Cremers, Bodo Rosenhahn, "Fusion of head and full-body detectors for multi-object tracking," *Paper*, 2017.
[11] Johannes Fitz, "Datenassoziation für multi-objekt-verfolgung mittels deep learning," *Paper*, 2020.
[12] Qi Chu, Wanli Ouyang, Hongsheng Li, Xiaogang Wang, Bin Liu, Nenghai Yu, "Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism," *Paper*, 2017.
[13] Q. C. W. O. H. L. X. W. B. L. N. Yu, "Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism," *ICCV*, 2017.
[14] T. M. A. K. L. L.-T. C. Feichtenhofer, "Trackformer: Multi-object tracking with transformers," *Paper*, 2022.
[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *NIPS*, 2017.
[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *ICML*, 2015.
[17] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, 2014.
[18] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *AISTATS*, 2010.
[19] I. G. Y. B. A. Courville, *Deep Learning*. MIT Press, 2019.
[20] D. M. R. N. V. S. Reddy, "Effect of padding on lstms and cnns," *Paper*, 2019.