Data-Driven Multi-Object Tracking: Integrating Modular Neural Networks into a Kalman Framework

Christian Alexander Holz, Christian Bader, Markus Enzweiler, Matthias Drüppel

***Abstract*—This paper introduces Machine Learning (ML)-based methodologies for Multi-Object Tracking (MOT) tailored to Advanced Driver Assistance Systems (ADAS). Our approach particularly addresses the growing complexity and precision requirements in automotive applications. We propose and evaluate three novel neural networks (NN) designed for MOT: (i) the Single Prediction Network (SPENT) for trajectory forecasting, (ii) the Single Association Network (SANT) for mapping single sensor inputs to existing tracks, and (iii) the Multi-Association Network (MANTa) for associating multiple sensor inputs to multiple tracks. These ML models are integrated with a traditional Kalman filter (KF) framework to enhance MOT performance while preserving the system's modularity and interpretability.**

In our evaluation, we compare the proposed methods against a traditional Kalman filter on the public KITTI tracking dataset: SPENT reduces the Root Mean Square Error (RMSE) by 50% on the KITTI tracking dataset compared to a standard KF; SANT and MANTa achieve a 95% validation accuracy in sensor object assignment to tracks. These results underscore the potential of integrating machine learning into tracking systems to enhance performance and robustness, while ensuring the modularity and maintainability of Advanced Driver Assistance Systems (ADAS) tracking systems, at the same time.

I. Introduction

THE ongoing evolution of Advanced Driver Assistance Systems has brought the need for precise and reliable Multi Object Tracking into the spotlight [1]–[8]. 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 already in the early timesteps of detection with a accurate prediction - a key challenge in Computer Vision 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 [1]–[3], [9], [10]. Tracking frameworks form the heart of ADAS that are used in millions of vehicles around the globe. The vast majority of these frameworks rely on classical approaches such as the KF or its variants [1]–[3]. 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 clean and comprehensible.

However, development for automated driving is highly complex [11] and in the automotive industry tracking systems are not developed for a single model, but rather as a platform software and are deployed to a variety of different vehicle 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 regularly leads to poor performance in certain scenes for a specific configuration. 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-engineering parameters and relying on heuristics leads to a software that is 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. (i) In contrast to the Kalman filter [3], 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 and habits of road users. Many conventional tracking systems rely on static methods for data association. Commonly used algorithms like the Hungarian algorithm (HA) [12] require heuristics and fixed thresholds. (ii) Our Single and (iii) Multi Association Network replaces the calculation of a distance metric for the corresponding assignment by employing machine learning in order to resolve situations unclear for traditional approaches. Both, our Single Prediction Network and our association networks

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. Enzweiler is with the Institute for Intelligent Systems, University of Applied Sciences, Esslingen, Germany

M. Drüppel is with the Center for Artificial Intelligence, Baden-Württemberg Cooperative State University (DHBW), Stuttgart, Germany

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 ADAS tracking systems by applying ML.

II. Related work

A. State prediction for tracked objects

One fundamental problem in Tracking-by-Detection 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], [2], [13]. However, they reach their limits in more complex scenarios, particularly in the presence of non-linear motion patterns and interactions among multiple objects [14]–[16]. Ristic et al. [14] discuss the limitations of Kalman Filters in nonlinear and non-Gaussian scenarios and introduces Particle Filters as an alternative solution. Julier et al. [15] propose the Unscented Kalman Filter (UKF) as an extension to the standard Kalman Filter to handle non-linear motion models more effectively. Wan et al. [16] introduce the Gaussian Mixture Model for tracking multiple objects in a cluttered environment. 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, which can enable more precise and flexible data-driven object state predictions.

B. Association of sensor objects to tracks

Another fundamental problem for a TbD tracker is the data association. One of the most common methods for data association is the Global Nearest Neighbor (GNN) algorithm. The GNN algorithm assigns the nearest detected object to the existing tracks based on a distance metric. Although GNN is often implemented using the Hungarian Algorithm [12] to find the optimal assignment between observations and tracks, it only considers the current set of observations and does not explicitly take temporal continuity or motion patterns into account. This limitation can lead to incorrect assignments in complex scenarios, especially when objects cross paths or when sensor measurements are noisy. The Hungarian Algorithm considers the assignment problem as a linear sum assignment problem taking into account the current state of the tracks and the sensor objects. Other approaches, such as the Joint Probabilistic Data Association (JPDA) [17], consider the temporal information of the tracks and the sensor objects. However, these methods are computationally expensive and require the calculation of a joint probability distribution for all possible assignments. ML-based approaches have been proposed to address these limitations. This aggregate of temporal information can be done for example using attention mechanisms as used in [8], [18] or recurrent neural networks (RNNs) as developed in [4], [5]. The latter is what we are also pursuing in this work. Similar to the problem statement by Mertz et al. [5], 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. 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 as is elaborated in the next chapter.

C. Real time applications

ADAS are embedded real-time applications where it is crucial to predict the state of objects immediately after their detection as described in [1], [19]. This rules out offline tracking methods as presented in [20] that process the entire data sequence at once in a batch process. Therefore, most recent approaches for tracking multiple objects rely on online methods that do not depend on future sensor 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: Our ML models are integrated into a Kalman Filter tracking system. The Kalman Filter is a popular choice for a multi-object tracker due to its interpretability and performance [1]–[3], [13]–[16]. Bewley el al. [1] present a simple and efficient multi-object tracking approach based on the KF and the Hungarian Algorithm. Seidenschwarz el al. [2] present a MOT approach based on simple visual cues. The authors argue that many existing multi-object trackers are overly complex and require significant computational resources. Instead, they propose a image-based and more straightforward approach that leverages basic visual features such as color, shape, and motion. These cues are used to performe image-level object tracking and to make frame-to-frame associations.

b) Recurrent Neural Network based Tracker: Similar to our methodologies, studies by [4], [5], [21] introduce RNN-based approaches for online multi-target tracking. Specifically, the work by Mertz et al. [5] focuses on data association within a TbD framework. Their proposed DeepDA model, an Long Short-Term Memory (LSTM)-based Deep Data Association Network, is designed to learn and execute the task of associating objects across frames. This model's ability to discern association patterns directly from data enables the achievement of robust and reliable tracking outcomes, even in environments with significant disturbances. Mertz et al. employ a distance matrix, derived from the Euclidean distance measure, as the input for the DeepDA network. This innovative approach effectively supersedes traditional association algorithms, such as the Hungarian Algorithm. It is inferred that the Euclidean distance measure served as a foundation not only for generating the ground truth (GT) training data (i.e., distance matrices) but also for the subsequent evaluation process. However, this is not explicitly stated. In our work, we want to enable the network to follow a completely data-driven association logic without forcing a concrete distance metric.

c) Attention Mechanism based Tracker: In our work we analyze tracks using LSTM based models. An alternative approach would be the use of the attention mechanism [22]

as e.g. applied by the authors of [6]–[8], [18], [23]. The main research focus of Hung et al. [8] is on soft data association (SoDA), which enables the tracker to make probabilistic associations between objects and account for uncertainties in the associations. Their approach to SoDA works by using the 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 Single Prediction Network (SPENT), (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 $Z_{t,1:m}$ in every time step. If new objects are detected, these are stored in a k-dimensional state vector containing information such as object position (x,y), yaw angle and object dimensions (length and width) (for $k = 5$). The Prediction Network predicts all vectors $X_{t,1:n}$ to the next time step where they are used as input to either the SANT or MANTa association networks. These provide the association matrix $A_{t,1:m}$, that is used to update the tracks $Tu_{t,1:n}$ 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 $Tc_{t,1:n}$ to the next higher software component that have been confirmed by sensor objects.

IV. Tracking with ML-based prediction and association networks

We apply the tracking-by-detection paradigm, in which a tracker fuses object detections to generate object tracks that are consistent over time. Vo et al. [13], 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 LSTM and bidirectional Long Short-Term Memory (BILSTM) networks. Both for the prediction and the association of the sensor objects to the existing tracks.

A. Single Prediction Network (SPENT)

In our methodology, we leverage the hidden states of the LSTM layer as a dedicated information repository for each object. The Prediction Network is architected for open-loop operation, signifying that it forecasts future state values based on data previously received. Within the context of an online MOT process, this enables the network to prognosticate the most probable subsequent state values by utilizing the state values received for a specific sensor object's state vector.

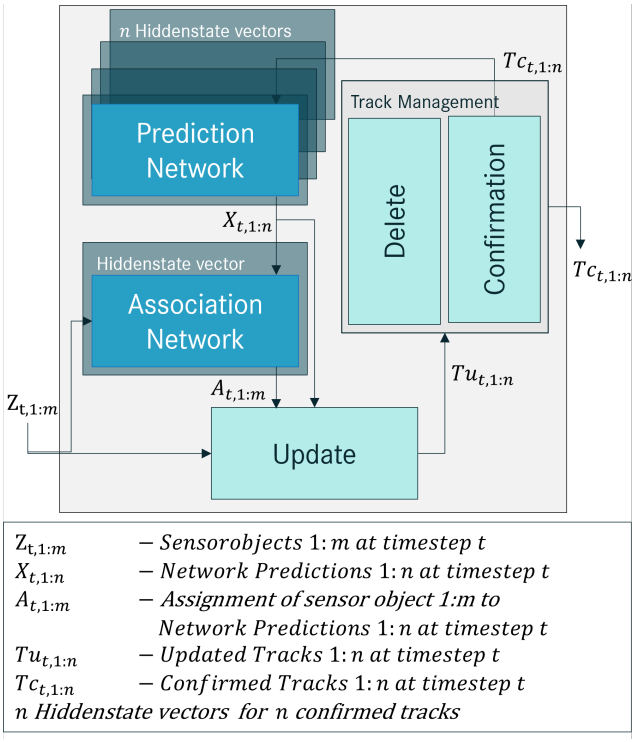

Fig. 1. Schematic representation of the two integrated networks (blue) in a tracking-by-detection framework. The Association Network can either be filled with our SANT (single) or MANTa (multi) association models.

a) Data preprocessing: In the development of our model, ground truth data comprising vehicle tracks (cars and vans) from the KITTI dataset was utilized. A GT track encapsulates the temporal evolution of an object, delineating its trajectory from the moment it enters until it exits the sensor's detection range. For preprocessing, a program was developed to extract all tracks of the selected object classes (cars and vans) per sequence and store them in a unified dataset. Through this iterative process, 635 tracks (cars and vans) were extracted from the entire KITTI dataset. Tracks with a short temporal length were filtered using a threshold of 3 frames, resulting in a final dataset comprising 624 tracks. The track lengths vary from a minimum of 4 frames to a maximum of 643 frames (Sequence 20, ID 12). To enhance model generalization and foster convergence during training, we normalized the state values of tracks at time t (predictors) and at time $t+1$ (targets) in accordance with the methodology outlined in [24]. This normalization process aimed to standardize the distribution of both predictors and targets to have a mean of zero and a unit variance. The mean value μ and standard deviation σ for each state variable were computed across all tracks, employing the subsequent equations:

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

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

In our approach, we incorporate pre-padding as delineated by Reddy et al. in [25], where the authors elucidate the effect

of padding strategies on the performance of neural networks in sequence-oriented tasks. Their investigation reveals that while both pre-padding and post-padding are viable options, the selection of padding technique plays a pivotal role in the model's efficiency. This is particularly salient for LSTM networks, where the contextual integrity of the sequence is paramount for optimal performance. 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.

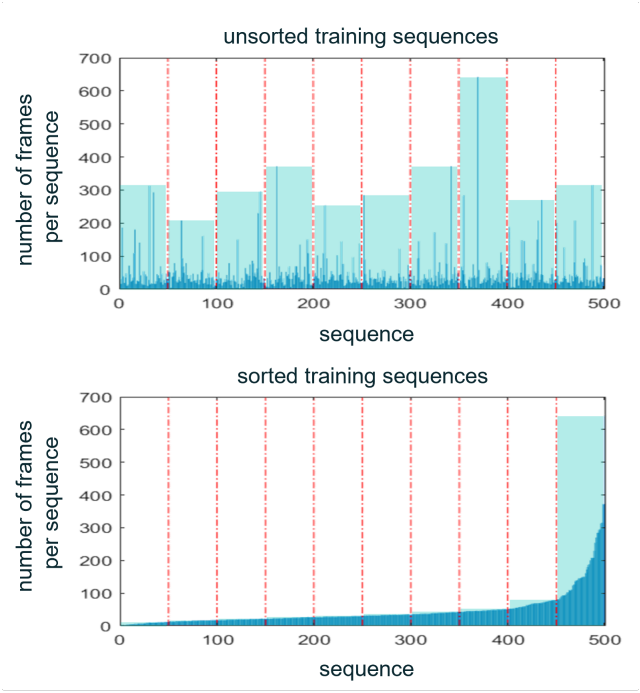

Fig. 2. Analysis of Sequence Padding: Unsorted vs. Sorted Data. Delineates the differential impact of padding on LSTM network training, contingent upon whether the input data sequences are sorted by length. In the upper panel, unsorted data necessitates extensive padding to equalize the sequence lengths within a batch, thereby augmenting the computational overhead. Conversely, the lower panel illustrates that sorting data by sequence length prior to batching significantly curtails the requisite padding.

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. As highlighted by [25], while padding introduces noise into the network, it is essential for aligning sequences within each mini-batch to facilitate the training of LSTM networks. To mitigate this noise, we sorted the sequences in the training dataset by their length prior to applying mini-batch padding. As demonstrated in Figure 2, this approach significantly reduces the padding (depicted in turquoise) required for each mini-batch by utilizing a pre-sorted dataset. This adjustment was critical for achieving convergence during the training process.

b) Network architecture: As depicted in Figure 3, the schematic illustration of the generic structure of the Prediction Network elucidates how the architecture is adeptly designed to address the challenges of real-time state prediction. This representation highlights the strategic deployment of the LSTM layer for storing and processing object-specific information, facilitating accurate and timely predictions of object states.

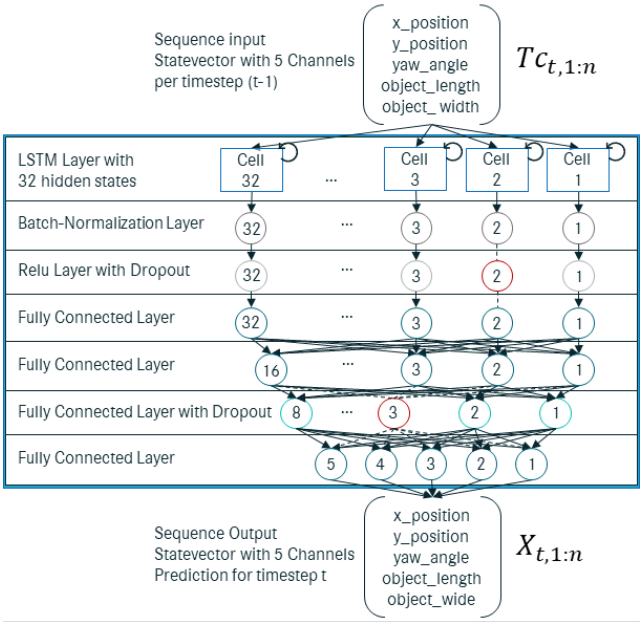

Fig. 3. Schematic representation of the generic prediction network structure.

The foundational layer of our Prediction Network is constituted by an LSTM layer, wherein the internal states — known as hidden states — are dynamically updated at each timestep in response to incoming measurement data. This iterative updating mechanism facilitates a continuous correction throughout the sequence of each track, thereby enhancing the predictive accuracy of the network. The quantity of hidden units within this layer directly correlates to the volume of information retained across timesteps, as illustrated in Figure 3. These hidden states are capable of encapsulating information from all preceding timesteps, independent of the sequence's length, ensuring a comprehensive temporal understanding. Subsequent to the LSTM layer, our architecture incorporates a batch normalization layer, which standardizes the LSTM output prior to its transition to the following Relu layer. This normalization significantly expedites the training process and fosters better convergence by mitigating internal covariate shift, as supported by [26]. Following this, a Relu layer is employed to apply a non-linear threshold operation, setting any input value below zero to zero. During training, a dropout layer is introduced to randomly nullify input elements with a specified probability, thereby imposing a regularization effect and preventing overfitting, as detailed in [27]. The architecture culminates in a Fully Connected (FC) Layer, which amalgamates the localized insights garnered by preceding layers. The dimensionality of the final FC layer is meticulously aligned with the number of response variables required by the output layer, as elucidated in [28].

Our model's loss function is based on the Mean Squared Error (MSE) metric, which is calculated for each state value prediction. The MSE quantifies the average squared discrep-

ancy between the predicted and actual target values. We chose MSE because it provides a clear and direct measure of how closely our predictions align with the true states, making it an effective metric for optimizing the accuracy of our model's state predictions.

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

where k is the size of the predicted state vector, y_i is the ground truth value (KITTI cars and vans 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 T. The Factor 1/2 simplifies the calculation of the gradient during backpropagation, making the derivative of the loss function with respect to the network's predictions easier to compute:

$$loss = \frac{1}{2T}\sum_{j=1}^{T}\sum_{i=1}^{m}(y_{ij} - \hat{y}_{ij})^2. \tag{3}$$

During training, the average loss is calculated using the observations in the mini-batch, so T equals the mini-batch length.

Within the context of the KITTI dataset, encompassing both cars and vans, our model attained a Root Mean Square Error of 0.029 for the positional prediction of all objects within the test dataset, representing 31 Tracks (see Table I). This performance metric translates to an average deviation of 0.42 meters for predictions pertaining to the X coordinate and 0.23 meters for those related to the Y coordinate. Using an inhouse implemented Kalman Filter carried by Daimler Truck Research Group following [1], [3], an RMSE of 0.066 was achieved on the same data set.

B. Single Association Network (SANT)

The association network facilitates a data-driven methodology to address the combinatorial optimization challenge of data association, which is known to be NP-hard. Unlike traditional association methods referenced in [4], [5], our SANT model innovates by not relying on a predefined distance matrix as input. Instead, it processes the extant tracks alongside each newly measured sensor object directly. SANT is designed to match a singular sensor object to one of multiple track objects. This approach obviates the necessity for a predefined distance metric, thereby endowing the network with the autonomy to devise its own association logic, which it learns from the training data. Consequently, the association network supplants the traditional computation of a distance metric and the implementation of an association algorithm, such as the Hungarian Algorithm, with a data-driven, learning-based methodology.

a) Data preprocessing: In the formulation of SANT, we conceptualized data association as a temporally structured challenge, adopting a sequence-to-vector paradigm. This framework posits the association of a singular sensor object, denoted as $Z_{(t,1)}$, with a collection of tracks, represented as $X_{(t,1:n)}$. These tracks were meticulously curated from the KITTI dataset, which comprises annotated camera recordings. It is imperative to note, however, that genuine ground truth data for the specific association problem at hand are not available. To guarantee the unambiguity of assigning a sensor object to a single track within a specified set of tracks, each sensor object was synthetically generated at a given timestep from the pre-existing track set of that timestep. Furthermore, to simulate realistic sensor data from the GT data, artificial noise was introduced. This process was meticulously designed to reflect the inherent inaccuracies and uncertainties present in real-world sensor measurements. 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. The actual number of tracks can vary between 0 and a maximum of 16 objects in relation to the KITTI data set and the selected objects (cars and vans). A comprehensive analysis of the data is presented in the MANTa section.

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 the one to-be-assigned sensor object and the currently multiple tracked objects.

Architectures with RNN, GNU, LSTM and BILSTM layers were tested. As part of these investigations, the best association performance was achieved with the BILSTM layer. The network shown in Figure 4 achieved the best performance in comparison with a Training Accuracy of 95% and Validation Accuracy of 95%. By combining the outputs of two LSTM layers that pass the information in opposite directions, [29] demonstrates the ability to capture the context from both ends of the sequence. The resulting architecture is called BILSTM. The output mode has been configured in the BILSTM layer, so that the layer is able to receive a sequence as input and output value vector. This form of dimension reduction is necessary in order to carry out a corresponding classification. The FC layer 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. The cross-entropy cost function is utilized to quantify the discrepancy between the network's probabilistic predictions and the ground truth values, a method particularly suited for tasks involving categorically exclusive classes. This approach employs one-hot encoding to transform class representations into binary vector formats, thus enabling the delineation of each class within a 1-to-n coding

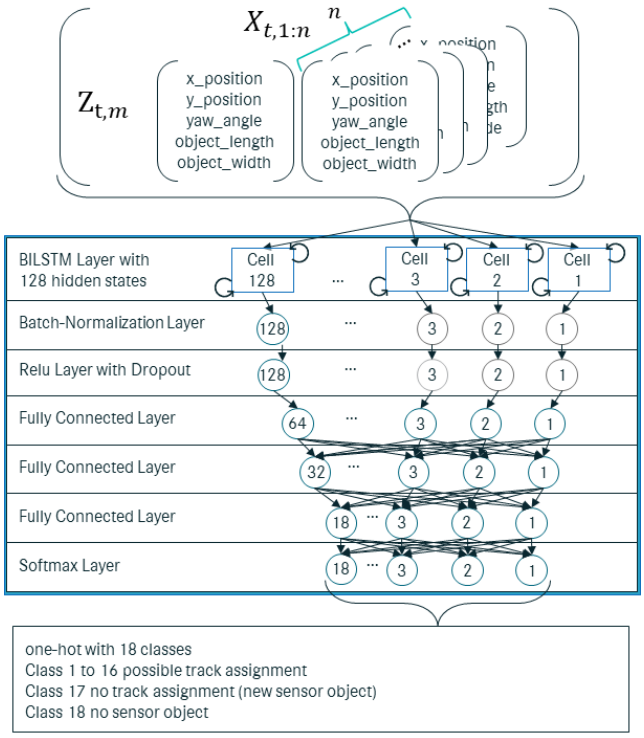

Fig. 4. Schematic representation of the generic network structure of SANT.

scheme. The cross-entropy loss for each prediction, relative to its corresponding target value, is computed as follows:

$$loss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} T_{ij}\log(Y_{ij}), \tag{4}$$

where N denotes the total number of samples, C represents the number of class categories, T_{ij} is the GT indicator for whether class j is the correct classification for sample i, and Y_{ij} is the predicted probability that sample i belongs to class j, as derived from the softmax function output.

C. Multi-Association Network (MANTa)

The development of the single association network demonstrated that data-driven association logic can be effectively learned by a deep learning model. Building upon this foundation, the objective was to create a network capable of associating multiple sensor objects (m SOs) with multiple tracks (n tracks) in a single operational step. Unlike the Single Association Network, MANTa is specifically designed to handle the simultaneous association of multiple sensor objects to multiple tracks. To address this complex task, MANTa was developed to solve the following association scenarios:

- **1 to n** - one sensor object (SO) with n tracks
- **m to 1** - m sensor objects (SOs) with one track
- **m to n** - m sensor objects (SOs) with n tracks
- **m to 0** - m sensor objects (SOs) with no tracks
- **0 to n** - no sensor objects (SOs) with n tracks
- **0 to 0** - no sensor objects (SOs) with no tracks

Being able to cover these scenarios enables MANTa to adeptly manage the complexities inherent in multiple object tracking (MOT), providing a robust and scalable solution for real-world applications. With the integration of MANTa into a Multi-Object Tracking 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 and capture information over time when no data is available.

a) Data preprocessing: The training, validation and test data set of MANTa was created according to the described objective. Figure 5 shows the input data structure with corresponding association tasks for time step 85 of sequence 20 of the KITTI data set. The respective tracks per time step were extracted across all sequences. All extracted tracks were each modified with noise as in the SANT development and then normalized. Figure 5 shows the non-noisy sensor objects to enable a example assignment (given by the one-hot vector at the bottom) and increase understanding of the association procedure. The values obtained in this way were arranged as measurements (sensor objects) in a pseudo-random order per time step in an $F_{total} \cdot T_{max}$ matrix. The one-hot vector shown in figure 5 shows the GT assignment of the first sensor object to the track at position two. Where F_{total} represents the total number of features and T_{max} stands for the maximum number of existing tracks per time step. The total number of features results from the status values per track and SO set. (here, $F_{total} = 2 \cdot m$, with $m = 5$ being the number of state values in this investigation). For the defined test case with cars and vans, the KITTI data set results in $T_{max} = 16$. There are seven 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).

Fig. 5. MANTa, data structure, shows the non-noisy sensor objects to enable a visual assignment and increase understanding of the association procedure. Seven tracks are extracted from the KITTI data set for the time step of sequence 20. Eight sensor objects are generated and pseudo-random ordered. The one-hot vector shows the GT assignment of the first sensor object to the track at position two.

The GT assignment is displayed at the bottom of the section. The assignment output can be thought of a matrix with dimensions maximum number of tracks $T_{max} = 16$ and the number of possible assignment classes $C = 18$. Where each field can either be zero (no assignment) or one (assignment). For numerical reasons, we unfold this matrix to a one-hot vector of dimension $OH_{max} = 288 = 16 \cdot 18$. 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.

As for SANT the cross-entropy cost function calculates the cross-entropy loss between network predictions $Y_{k,i}$ and target values $T_{k,i}$ for the unique assignment task for mutually exclusive classes. The already introduced one-hot vector is used to represent the class in binary form in a vector and thus generate a 1-to-OH_{max} code. The following formula is used to calculate the cross-entropy loss values for each time step t:

$$loss_t = \sum_{i=1}^{OH_{max}} \left(T_{n,i} \ln(Y_{n,i}) + (1 - T_{n,i}) \ln(1 - Y_{n,i})\right) \quad (5)$$

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

$$loss = -\frac{1}{X} \sum_{xx=1}^{S} loss_t \quad (6)$$

MANTa is trained to assign a list of sensor objects (SOs) to a list of tracks in a single operational step on the same dataset as SANT and achieved an assignment accuracy of 95% for the six most frequently occurring association sets. Thus are sets with one to six tracks per time step.

In the context of the entire KITTI dataset, which includes both cars and vans objects, MANTa achieves an average association accuracy of 70%. This limitation was primarily attributed to the characteristics of the extracted data. As illustrated in Figure 6, the distribution of the numberof existing tracks per time step reveals significant insights. Notably, time steps containing at least one track constitute nearly one-third of the entire dataset, accounting for 29.9% of the data, which corresponds to an absolute count of 2315 samples. Time steps containing one to six tracks constitute 81.5% of the samples and are therefore 6374 of 7827 samples. Consequently, tests were conducted using a reduced dataset with one to six tracks per time step to demonstrate the multi-association capability of the network. MANTa correctly assigns 95% of the dataset for time steps containing one to six tracks. This result confirms MANTa's proficiency in handling data with the appropriate dataset, as SANT also achieves a validation accuracy of approximately 95% across the entire KITTI dataset (including cars and vans). The primary advantage of MANTa over SANT

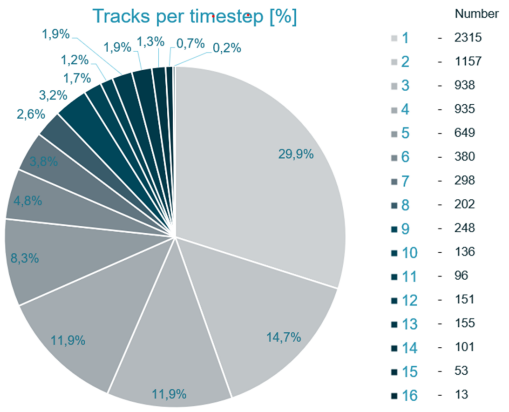

Fig. 6. The diagram shows the distribution of the number of existing tracks per time step. For the KITTI dataset, which includes both car and van objects. It reveals that time steps containing one to six tracks constitute 81.5% of the samples (6374 of 7827). The ledgend shows the absolute number of samples which contain the respective number of tracks. In 29.9% of the samples, the data contains one track, in 14,7% two tracks. Only 18.5% of the samples contain more than six tracks, which leads to an unbalanced dataset (1448 of 7827 samples).

is its ability to assign multiple sensor objects to multiple tracks in a single operational step.

b) Network architecture: The schematic representation 7 shows the developed network architecture 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.

The BILSTM layer processes the input data as already explained for SANT. The task of associating a sensor object list with a track list requires a separate network part for each track. This extension is labelled accordingly in figure 7. For each track (from 1 to $T_{max} = 16$), the MANTa has been developed with the fully connected, softmax stack that was introduced for SANT. Each softmax output consists of a vector with $C = 18$ elements,n which represents the most probable assignment. This means that a single assignment can be realized 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.

V. Conclusion

The developed networks can be modularly integrated into the Tracking-by-Detection framework, effectively replacing classical heuristic algorithms. SPENT substitutes the state predictions of the Kalman Filter. The trained network estimates predictions per time step without the necessity of a dynamics model. Our implementation uses a recurrent network to update its internal hidden states at each time step, thereby achieving accurate state predictions without external correction. The model is suitable for real-time applications and represents a viable alternative to classical prediction methods. Network

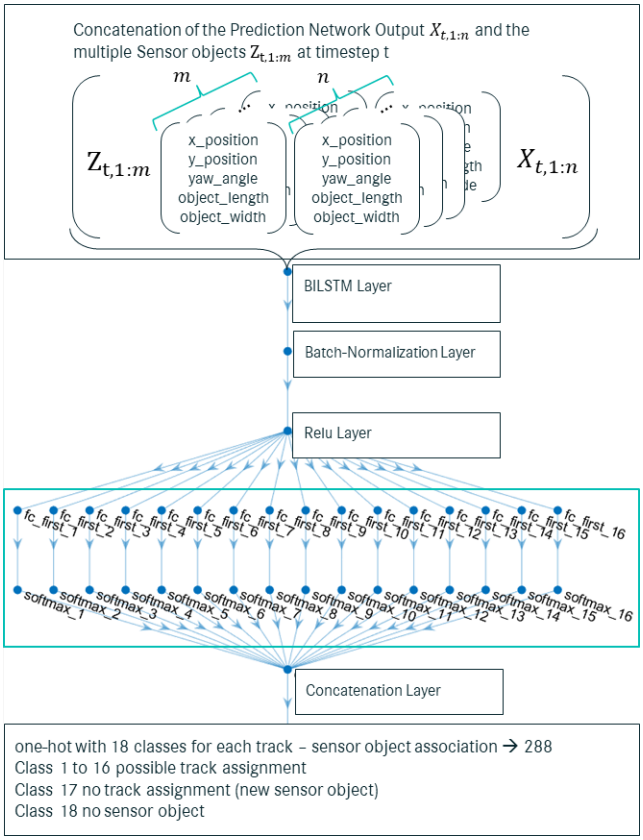

Fig. 7. Schematic representation of the generic network structure of MANTa.

verification demonstrates a Root Mean Square Error (RMSE) of 0.027, averaged over the training, validation, and test datasets as shown in Table I. Translated in meters this error corresponds to an average deviation of 0.42 meters for the X coordinate position and 0.23 meters for the Y coordinate position predictions, both relative to the ego vehicle.

TABLE I
COMPARISON OF RMSE VALUES FOR THE SPENT AND KALMAN FILTER IMPLEMENTED BY DAIMLER TRUCK RESEARCH GROUP. THE KITTI DATASET FILTERED FOR CARS AND VANS OBJECTS WAS USED AND SPLIT INTO TRAINING, VALIDATION, AND TEST DATASETS. 90% OF THE DATA WITH 624 TRACKS WAS USED FOR TRAINING, 5% FOR VALIDATION, AND 5% FOR TESTING.

– Dataset – Number of tracks	**Training** 562	**Validation** 31	**Testing** 31
Model	**RMSE**		
SPENT	0.025	0.027	0.029
KF	0.066	0.065	0.066

For another primary task of the TbD MOT method, data association, SANT was developed as a replacement for the classical Global Nearest Neighbor (GNN) method. SANT can replace algorithms for calculating a distance metric and assignment procedures such as the Hungarian Algorithm (HA) with a learned, data-driven assignment logic. Based on a defined validation dataset with approximately 2700 samples, SANT achieves an accuracy of 95%. MANTa is an advancement of SANT and addresses the limitation of individual assignment. This network extension is capable of assigning a set of sensor objects (m) to a set of tracks (n). Detailed data analysis identified limitations affecting network performance. Verification results indicate that MANTa achieves an assignment accuracy of 95% for the six most frequently occurring association sets.

ACKNOWLEDGMENTS

We would like to express our sincere thanks to the organizations that provided financial support for this research project, namely the Daimler Truck AG and the Baden-Württemberg Cooperative State University (DHBW) Stuttgart.
[**Markus:** In some references, the author names are wrongly abbreviated.]

REFERENCES

[1] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Sort: Simple online and realtime tracking," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 3464–3468.
[2] Jenny Seidenschwarz, Guillem Brasó, Victor Serrano, Ismail Elezi, Laura Leal-Taixé, "Simple cues lead to a strong multi-object tracker," *Paper*, 2022.
[3] J. Krejcí, O. Kost, O. Straka, and J. Duník, "Bounding box dynamics in visual tracking: Modeling and noise covariance estimation," *2023 26th International Conference on Information Fusion (FUSION)*, pp. 1–6, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:261126559
[4] Anton Milan, Seyed Rezatofighi, Anthony Dick, Ian Reid, Konrad Schindler, "Online multi-target tracking using recurrent neural networks," *Paper*, 2016.
[5] H. L. H. Z. C. Mertz, "Deepda: Lstm-based deep data association network for multi-targets tracking in clutter," *Paper*, 2019.
[6] 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.
[7] T. M. A. K. L. L.-T. C. Feichtenhofer, "Trackformer: Multi-object tracking with transformers," *Paper*, 2022.
[8] W.-C. H. H. K. T.-Y. L. Y. C. R. Yu, "Soda: Multi-object tracking with soft data association," *Paper*, 2020.
[9] A. Kampker, M. Sefati, A. S. A. Rachman, K. Kreisköther, and P. Campoy, "Towards multi-object detection and tracking in urban scenario under uncertainties." in *VEHITS*, 2018, pp. 156–167.
[10] H. Wu, W. Han, C. Wen, X. Li, and C. Wang, "3d multi-object tracking in point clouds based on prediction confidence-guided data association," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 5668–5677, 2022.
[11] S. D. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang, "Perception, planning, control, and coordination for autonomous vehicles," *Machines*, vol. 5, no. 1, 2017. [Online]. Available: https://www.mdpi.com/2075-1702/5/1/6
[12] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, 1955.
[13] Ba-Tuong Vo, "code set for research use: Multi-sensor multi-target tracking," *Code*, 2013. [Online]. Available: https://ba-tuong.vo-au.com/codes.html
[14] B. Ristic, S. Arulampalam, and N. Gordon, *Particle filters for tracking applications*. Artech House, 2004.
[15] S. J. Julier and J. K. Uhlmann, "The unscented kalman filter for nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.
[16] E. A. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC)*. IEEE, 2000, pp. 153–158.
[17] J. Dezert and Y. Bar-Shalom, "Joint probabilistic data association for autonomous navigation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 29, no. 4, pp. 1275–1286, 1993.
[18] 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.

[19] N. Wojke, A. Bewley, and D. Paulus, "Deepsort: Simple online and realtime tracking with a deep association metric," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3645–3649.
[20] Roberto Henschel, Laura Leal-Taixé, Daniel Cremers, Bodo Rosenhahn, "Fusion of head and full-body detectors for multi-object tracking," *Paper*, 2017.
[21] Johannes Fitz, "Datenassoziation für multi-objekt-verfolgung mittels deep learning," *Paper*, 2020.
[22] 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.
[23] 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.
[24] I. G. Y. B. A. Courville, *Deep Learning*. MIT Press, 2019.
[25] D. M. R. N. V. S. Reddy, "Effect of padding on lstms and cnns," *Paper*, 2019.
[26] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *ICML*, 2015.
[27] 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.
[28] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *AISTATS*, 2010.
[29] S. H. J. Schmidhuber, "Long short-term memory," *Neural Computation*, 1997.