A data-driven approach for pedestrian intention estimation

Benjamin Völz[1], Karsten Behrendt[2], Holger Mielenz[1], Igor Gilitschenski[3], Roland Siegwart[3], and Juan Nieto[3]

***Abstract*— In the context of future urban automated driving many important problems remain unsolved. A critical one is the analysis and prediction of pedestrian movements around urban roads. Especially the analysis of non-critical situations has not received much attention in the past. This paper focuses on analyzing and predicting movements of pedestrians approaching crosswalks, a very crucial pedestrian-vehicle interaction in urban scenarios. In our previous work, we analyzed the performance of a data-driven Support Vector Machine-based architecture, and the relevance of specific features to infer pedestrian crossing intentions. In this paper, we will use our previous results as baseline to compare against an architecture based on neural networks for time-series classification. In particular we analyze the effectiveness of dense and Long-Short-Term-Memory networks. Furthermore, we will be looking into enhancing our feature vectors by adding LiDAR based images to the classification process. Additionally the evaluation provides an estimate for the temporal prediction horizon. The approaches presented are validated with real world trajectories recorded in Germany. Our results show an average accuracy improvement of $10-20\%$ with respect to our previous Support Vector Machine-based approach.**

I. Introduction

Predicting the movement of arbitrary objects is a crucial part of automated driving systems. When considering urban automated driving especially the long-term prediction of pedestrian trajectories represents a major challenge. To illustrate this, consider the example shown in Figure 1 where a car and a pedestrian approach a crosswalk. The car is obliged to stop if the pedestrian intents to cross the street. Timely inference of pedestrian intentions is extremely difficult, and designing a system for this requires important considerations from the vehicle's perspective. First, we do not want to execute an emergency braking maneuver or apply any sudden speed change. These actions would both be uncomfortable for the occupants of the car and highly dangerous for the pedestrian and other vehicles in the area. Second, we do not want to stop when unnecessary, i.e. if the pedestrian does not intent to cross the road. Late and false predictions in such situations will lead to a low system acceptance, apart from deteriorating traffic conditions. Additionally we also have to consider the pedestrians movement possibilities. Although the speed of pedestrians is in general much lower than the one of vehicles, pedestrians are much more agile. A pedestrian can change directions very quickly, for example by doing a sharp (e.g. $90°$) turn without reducing the speed. This high agility is what limits current systems to achieve reliable pedestrian movement predictions for only a few hundred milliseconds (e.g. [1]). Motivated by these fundamental problems, our work aims to develop a system that (i) minimizes false detections and (ii) provides long-term predictions to ensure smooth and safe maneuvers.

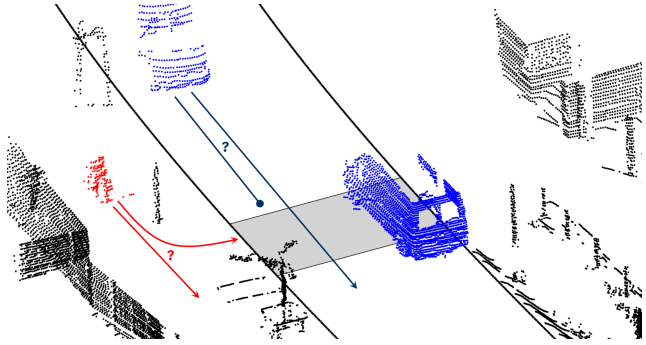


Fig. 1. Typical urban scenario: a car (blue) and a pedestrian (red) are approaching a crosswalk (grey box), where the pedestrian has priority. The inference problem involves realizing whether the pedestrian intents to cross the road. For the (e.g. automated) car this information is vital to decide whether it has to stop before the crosswalk or not.

One of the main findings of our previous work [2] was the difficulty to build hand-crafted features that generalize well. Deep learning architectures are able to provide end-to-end learning, obtaining therefore the features from the data. This property motivated the idea of utilizing a deep-learning architecture for the inference of pedestrian intentions. In this paper we will focus on the prediction of the pedestrians' intention to cross the street at a given crosswalk. In our previous work, we utilized a Support Vector Machine (SVM) based pipeline with very good results for the given problem. We use this pipeline as a baseline for our comparison of different neural network architectures. In this paper, we will first introduce a dense neural network for a fixed number of time-steps and features to directly classify a pedestrian's intent. For this we will use exactly the same input for both the neural network and the SVM.

In addition to the dense network, a Long-Short-Term-Memory (LSTM) network is created to allow time-series inputs of different sizes. Since LSTMs have been created for learning in time series [3] we expect a higher accuracy. A few optimal features could be created by capturing video feeds of the pedestrians, such that future poses and orientation could be inferred from images. Unfortunately, our current dataset does not contain that information. Therefore we created 2d images from LiDAR data. The Velodyne LiDAR provides

[1]Benjamin Völz and Holger Mielenz are with Corporate Research, Robert Bosch GmbH, 71272 Renningen, Germany benjamin.voelz@de.bosch.com

[2]Karsten Behrendt is with Chassis Systems Control, Robert Bosch LLC, Palo Alto, CA 94304, USA

[3]Igor Gilitschenski, Roland Siegwart, and Juan Nieto are with the Autonomous Systems Lab, ETH Zurich, 8092 Zurich, Switzerland

a range and an intensity value for every sampled point. These images allow us to gather information from pose and change in pose over time and possibly let us infer information for our problem. For each point, the id of the recording laser, as well as the rotation angle of the LiDAR itself, are known. With these known angular coordinates, it is possible to create 2D images for each spin, for example by coloring by intensity or range [4]. This way, the remarkable image processing classification capabilities of convolutional neural networks may be leveraged. A network is created to classify predictions solely based on images and another one in combination with our hand-crafted features.

The evaluation is performed on pedestrian trajectories recorded in Stuttgart, Germany, and features an evaluation of the temporal prediction horizon.

The specific contributions of this paper are:

- the formulation of a dense and a LSTM network for predicting pedestrian intention near crosswalks,
- a comparison between the different networks and baseline SVM,
- a performance analysis based on LiDAR-based 2D images,
- evaluation of the temporal prediction horizon.

The remainder of the paper is structured as follows: The state-of-the-art on predicting trajectories, behaviors and intentions of pedestrians in urban traffic is reviewed in Section II. Section III introduces different types of neural networks for classifying time-series of feature vectors. This includes the introduction of a convolutional network for image processing. The evaluation in Section IV first introduces the dataset, the hand-crafted features and gives an overview on our LiDAR-based 2D images. After that, the different types of networks are evaluated and compared to the SVM baseline. The conclusion is presented in Section V.

II. Related Work

In this section, we focus on the related work for both pedestrian path prediction and intention recognition. Recent research is primarily concerned with short-time vision-based pedestrian path predictions. These predictions are typically used for pedestrian protection systems and are therefore mostly designed to predict whether a pedestrian is going to stop at the curb or not (e.g. [1], [5], [6]).

Most of the vision-based algorithms combine both the detection and prediction of pedestrians. For the scope of this paper we will only analyze the different path prediction techniques and the features employed. An interesting study, that identifies which information human drivers use to decide whether a pedestrian will stop at the curb or not, is presented in [7]. They have shown that at least one part of the human body, either the head, the upper-body, or the legs, must be visible for a human driver to make correct predictions for the pedestrians' future movements. Consequently there has been a large number of work employing human body features. The most relevant work is reviewed in the next paragraphs.

The contour of the pedestrians' motion is used in [8] to infer their intention to cross the street. This contour includes implicitly the modeling of specific body language traits. In this case the main contributing features are the body bending and the spread of the legs. Similar approaches are presented in [5]. They show methods based both on the dense optical flow, and a low-dimensional flow-based histogram. They calculate so called motion features, which again capture both the leg and upper-body movement. These features are then linked with the pedestrians' position to create a special trajectory representation. These enriched trajectories are then used for trajectory matching. A larger variety of body parts, e.g. including arm movements, together with a sparse geometrical representation, where every body-part is depicted with a single line, is used in [9]. A common limitation of all discussed algorithms is the prediction horizon. For the given scenario (usually collision avoidance), the prediction accuracy is generally very high for a time horizon of only several hundred milliseconds. This value, however, is not enough for our application. Additionally the shown scenarios only review pedestrians who are approaching the street orthogonally.

One very important feature is missing from the previously shown approaches: the pedestrians' head orientation. A sophisticated approach is presented in [6]. Here the head orientation is used to determine the pedestrians situational awareness, i.e. if the pedestrian is aware of the approaching car. The paper incorporates this measure into a Dynamic Bayesian Network (DBN) and shows the benefit which a head tracking could add to existing prediction algorithms. They are able to outperform more complex state-of-the-art algorithms but still have a very limited time horizon.

Apart from these vision-based systems there are other interesting approaches that utilize the pedestrians' trajectory in terms of e.g. Cartesian coordinates in a specified coordinate frame. Again in the context of collision avoidance systems, [1] models the trajectory of the pedestrian together with the approaching car to analyze their remaining time to collision (TTC) with a Bayesian Network (BN). Additionally, concerning pedestrians in an arbitrary given environment, Gaussian process regression has been used to model pedestrian trajectory patterns [10]. These patterns represent the most common paths in this specific environment. A long-term prediction approach is presented in [11]. In a given urban environment hand-labeled goals for pedestrian movements are defined and used together with a jump-Markov process to model their behavior.

This paper aims to provide an approach able to provide predictions with longer time horizon which enables safer interaction between pedestrians and vehicles and is a basic requirement for fully automated driving systems.

III. Neural Network Architectures

This section presents the different architectures that we will evaluate. The demonstrated power of generalization of deep neural network architectures combined with its flexibility in building features are our main motivation to opt for this type of paradigm. In Section III-A we introduce a simple dense (feed-forward, fully-connected) network, which we use

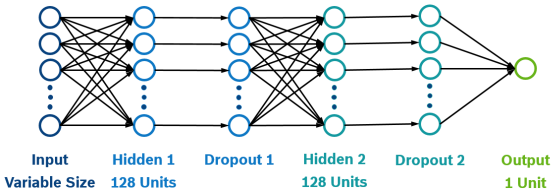


Fig. 2. Sample dense neural network with 2 fully connected layers, 2 dropout layers and a decision layer with sigmoid activation.

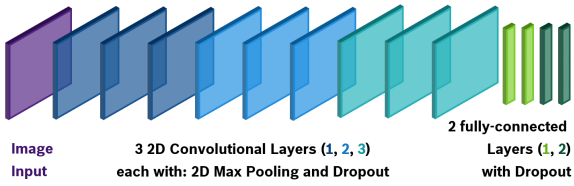


Fig. 3. A sample convolutional neural network. The Figure shows three convolutional layers, each followed by a max pooling and a dropout layer. The last convolutional layer is connected to two fully-connected layers.

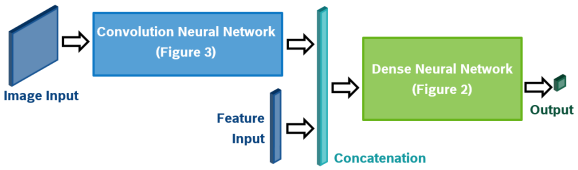


Fig. 4. Combination of the networks from Figures 2 and 3. The resulting network uses both features, the ones learned from image data and the hand-crafted features presented in our previous work to solve the given classification problem.

to create a neural network baseline. It also is the easiest network for initial tests since we can directly use the existing data without any changes. A more sophisticated network structure is presented in Section III-B. Recurrent networks are designed for learning in time series. Since our database consists of trajectories this matches our scenario perfectly. Furthermore we use convolutional networks (Section III-C) to learn features from our image source (compare Section I), these features can be used as either sol or additional input for any of the other networks. All networks are trained for the same classification task (intention recognition) with slightly different properties and inputs. Hyper-parameters were selected by searching within a hand-crafted set of options and then fine-tuning those.

A. Dense Neural Networks

Dense neural networks represent the straightforward approach of dealing with the classification of feature vectors. A dense neural network can be divided into several layers. In the case of feed-forward networks, each layer has a predefined number of neurons which are only connected to neurons in the next layer. All dense networks employed in this paper are similar to the depiction in Figure 2.

The input data, our feature vector, leads into a fully connected layer. Rectified linear functions [12] are used as activation function to attain some non-linearity and training stability. The activation layer is followed by a dropout layer [13] for regularization. This combination of fully-connected, activation and dropout layer is repeated a few times. The final fully-connected layer only has a single output neuron for classification which a sigmoid function transforms to values between -1.0 for not crossing the street with a very high probability and 1.0 for crossing.

B. Recurrent Neural Networks

Time-series data can often be analyzed more accurately using recurrent neural networks which allow feeding data back into previous layers. One widely employed variant contains Long-Short-Term-Memory (LSTM) units [3]. These networks store state information in their cells which is changed based on new inputs and previous outputs. The output is calculated based on cell state and input values.

LSTM networks are a combination of their cell state and four gate layers. Each gate represents a fully-connected layer with a fitting activation function that takes a concatenation of the current time-steps data and previous output as input. Those gates are then combined by element-wise multiplication and addition to a complete LSTM layer. The forget gate can decrease values in the cell, while the input and cell gates leads to an increase in values. The output is calculated by the output gate which decides which values are being used for classification in this case.

C. Convolutional Neural Network

The intent classification may also be possible using LiDAR-based images which can be analyzed using convolutional neural networks [14]. Image features are extracted by convolving trained filters along the image and using those features to classify the respective images. A first approach is done by only using image features and as a second step, the input vectors of our previous networks are added to the input. This feature combination happens at a later stage of the network by simply concatenating image features with the pre-calculated vectors. For regularization purposes, dropout layers are again added to the network. The basic network structures are outlined in Figures 3 and 4.

IV. Evaluation

For our evaluation we first provide an overview of our dataset. Afterwards we present a comparison between our previous SVM based classification results and the different neural network architectures from Section III. All neural networks were implemented in Python using Theano [15] and Lasagne [16]. Training is performed with AdaDelta [17] optimized stochastic gradient descent.

A. Dataset

As mentioned in [2], our database contains car and pedestrian tracks recorded with a Velodyne laser scanner. The raw point cloud is processed according to [18]. This includes the segmentation of the point cloud into arbitrary objects, the tracking of these objects over time and a classifier that issues one of four class labels: car, pedestrian, bicyclist or

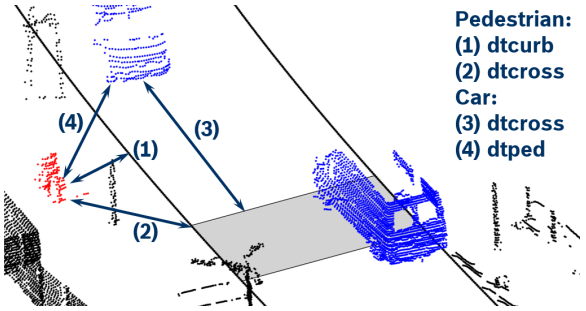


Fig. 5. Example of a Velodyne point cloud with an underlying sketch of the street. The two black lines mark the curbs and the grey box symbolizes the position of the crosswalk. The image contains the following Objects: cars (blue), pedestrians (red) and background (black). The image also shows a set of geometrical features which represent the objects movement relative to the crosswalk and relative to each other. Please note the the term "dt" is used as an abbreviation for "distance to".

background. The classifier consists of a nonlinear multiclass SVM trained and validated on the Stanford Track Collection (STC). Further details can be found in [18]. Figure 5 shows a preprocessed point cloud. Every track is associated with a precise digital map, which describes the static, urban environment, i.e. road boundaries, crosswalks and more. Altogether we use around 2000 trajectories with 100000 data points in this paper.

B. Hand-crafted Features and Automatic Labeling

Our previous work [2] presented a feature design and through analysis of them, therefore we will only provide a brief summary here.

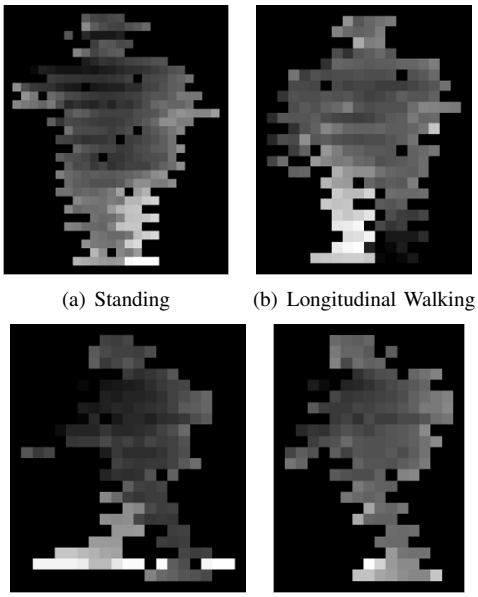
Our features can be separated into two main groups. The first group contains all features that only solely relate to the pedestrian. These features are: the velocity both in 2d coordinates and as an absolute value. The distance traveled in the previous time step and two distance measures, which describe the pedestrians' position relative to the road. *dtcurb* describes the orthogonal distance to the closest road boundary (usually a curb). The second distance measure is the minimal distance to crosswalk *dtcross*. This value will also be used in this section to provide insight on the prediction horizon. All geometrical features are shown in Figure 5.

The second feature group contains features based on the interaction of the pedestrian and a car. These features describe both the movement and position of the car (e.g. with a velocity and a distance to the crosswalk) and the "true" interaction in terms of a relative velocity and a distance between the pedestrian and the car.

Altogether this sums to 15 single features. These features are only suited to describe a single frame. To encompass temporal information we used the features from the last 4 frames as additional input for our machine learning algorithms. The total number of features sums up to 75.

In this paper we want to predict the pedestrians' intention to cross the street at a given crosswalk. Since our database contains the whole pedestrian trajectories, and our calculations are performed offline, we are able to automatically infer their intentions based on the observed movement. I.e. a pedestrians' trajectory is marked as *crossing* if we actually saw her crossing the street.

Please note that this method of automatic labeling has some disadvantages, which mainly arise due to sudden or severe motion changes. We have discussed these problems intensively in our previous work [2].



(a) Standing (b) Longitudinal Walking

(c) Lateral Walking (1) (d) Lateral Walking (2)

Fig. 6. Velodyne 2D image samples. The images show the Velodyne raw range measures color coded with a gray map (lighter colors correspond to smaller range measurements). The background has been removed from all images.

C. Image Data

For our first experiments we use the Velodyne [2]. This decision was made mainly because of it's 360° field-of-view and the availability of reliable object detection and tracking algorithms. Accordingly, raw LiDAR data are available for every track. The Velodyne provides for every point both the *id* of the measuring laser and the rotation angle of the sensor itself. Using these two information it is possible to create a 2D image in angular coordinates. Both of the Velodynes raw measurements (range and intensity) can be used to create gray scale images if plotted with a gray color map. For our purpose we use the previously mentioned object detection to both cut the pedestrians from the gray scale range image and remove the background. Same examples of the resulting images are shown in Figure 6.

D. Neural Network Training and Results

For our test we separate the database into a training (80%) and a test (20%) set. In this section we use only the training set for cross validation. Initial tests have shown that using all recorded features (with input dropout) does not improve results compared to the selected, minimal feature set (a subset of all features, from [2]). Most additional features led to fast over-training without improvements on the validation set.

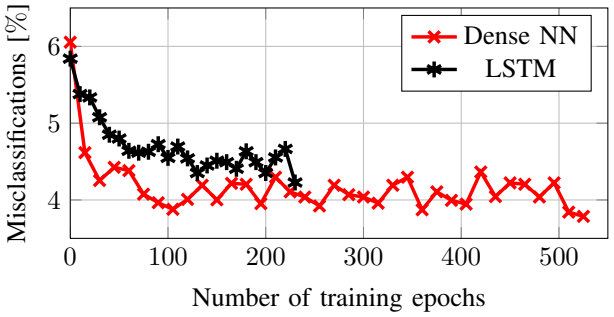


Fig. 7. Training progress of the Dense Network and the LSTM are shown over the number of training epochs for one training run of the cross-validation.

Most hyper-parameters were chosen by training several hundred different networks and selecting and fine-tuning the ones with the highest accuracy. The best performing dense neural network consists of three layers, each with rectified linear functions and dropout of 50%, and achieved a averaged cross-validation accuracy of about 96.21%. The number of units per hidden layer were 32, 64, and 128 . Figure 7 displays the training progress over time for one training run within the cross-validation.

The recurrent network did not achieve the same level of accuracy as the simple dense networks. Our best performing LSTM, a two layer LSTM with 64 and 128 hidden units, has a 95.77% cross-validation accuracy. LSTMs outperform when information has to be stored for a longer period of time. For pedestrians crossing the street, information about orientation and velocity from a few time-steps ago does not seem to be useful anymore. Usually, there is a, more or less, clear point where the pedestrian starts going towards the crosswalk but no prior information in their movements before that point. The advantage of the dense network is that it has simultaneous access to all currently relevant time steps and can make its decision based on all of those at the same time.

The convolutional networks did not offer additional insights into the pedestrian classification. Without the hand-crafted features, we could only achieve a 3.5% increase in classifying an input of images from 5 time-steps at a time over selecting the bias value. Adding image features to our hand-crafted input vector did not lead to any information gain. A detailed analysis of this will be given in the following section.

E. SVM vs. Neural Networks

In this section we will analyze the performance of our dense classification network from Section III compared to the SVM from our previous work [2]. This evaluation is performed on the test set introduced in the previous section. Figure 8(a) shows the percentage of correctly identified crossing pedestrians as a function of the distance to the crosswalk. All methods show an equally good performance for distance smaller than 3m. For all larger distances the simple dense neural network outperforms the SVM by 10 to 20%. This shows the potential of neural networks for identifying crossing pedestrians at large distances. For the combination of our hand-crafted features and the image-based features we did not obtain the expected improvement in performance. For most cases the performance is either identical or slightly worse than without the images. We assume that the major reason for this is the quality of the images. Although the Velodyne provides a 360° surround view, neither the horizontal nor vertical resolution provide detailed enough information. Usually it is possible to count the single pixels in one of these images (compare Figure 6), and especially at large distances it is possible that a pedestrian only consists of 20-40 points. Since it has been proven that image-based features can be used to improve the performance of state-of-the-art algorithms (e.g. [6]), we assume that we could achieve a better performance with a more detailed image source.

F. Evaluation of the Time Horizon

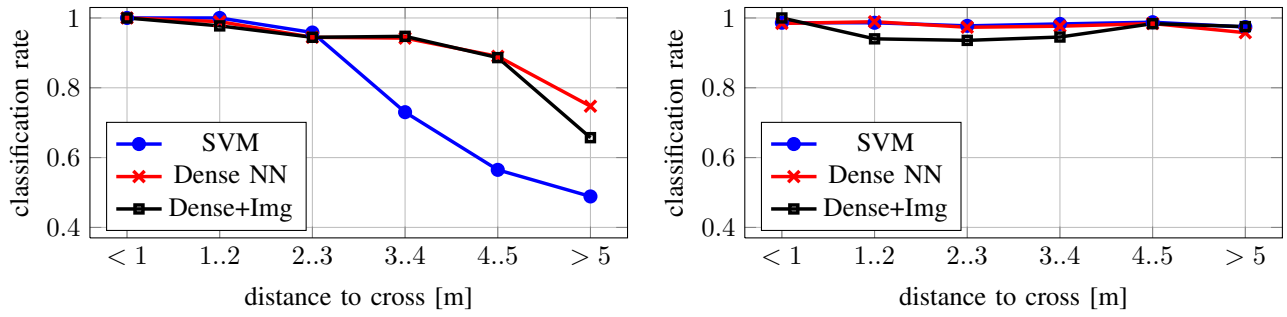
Usually pedestrians predictions around urban street are evaluated in respect to the remaining time-to-cross (e.g. [6]). This makes it possible to specify a temporal prediction horizon. Unfortunately, this procedure cannot be directly applied to non-crossing pedestrians. Their trajectories obviously do not cross the street and in many cases do not come close to it. Therefore it is not possible to estimate a time-to-cross for these pedestrians. Considering this together with the results from Figure 8, we decided to only analyze the crossing trajectories in this section. This means that the model is still trained with the full dataset, but the only the crossing trajectories from our test set are analyzed.

The results for our best dense neural network compared to the SVM are shown in Figure 9. We can see the limitations of our SVM baseline. Mainly due to vast speed changes the classification accuracy drops very fast even for small times (< 3 s). On the other hand we can see a totally different behavior for our dense neural network, where the accuracy is never lower than 80%.

Compared to our previous distance based evaluation (Figure 8(a)) we notice that the shown minimum accuracy is higher. The reason for this is easily explained: The highest observed time-to-cross in the shown portion of the database is 12 s. These high times correspond to a distance-to-cross > 5 m and belong to very slow walking pedestrians. Unfortunately the number of trajectories for such large times is relatively low in our current database. Therefore we decided to not evaluate the accuracy for these times. Hence for this time evaluation the slow walking pedestrians are biased by faster ones.

V. Conclusion

In this paper we proposed the use of deep learning architectures for identifying the pedestrians' intention to cross the street at a given crosswalk. First, we introduced a dense neural network which classifies intention based on features from several timesteps. Second, the time-series features are analyzed using recurrent networks, namely LSTMs. Third,



(a) Crossing pedestrians

(b) Non-crossing pedestrians

Fig. 8. Classification results for different network structures compared to the baseline SVM. The accuracy is shown both for crossing (a) and non-crossing (b) pedestrians. The shown neural networks are: the dense network solely with hand-crafted features (Dense NN), and with additional convolution layers for feature extraction from LiDAR images (Dense+Img). For better readability the results are evaluated relative to the discretized distance to cross.

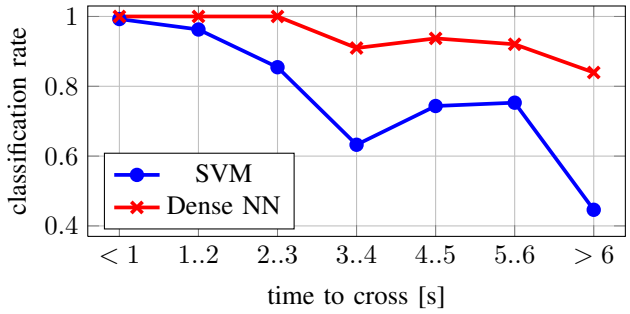


Fig. 9. Time-based evaluation. The results of both the SVM and the best dense neural network are shown. The classification accuracy is only evaluated for crossing pedestrians relative to their remaining time-to-cross. For better readability the time-to-cross is evaluated for discretized intervals.

the influence of image-based features learned from LiDAR images is analyzed. We have shown that all algorithms are able to outperform the baseline SVM. The best results are achieved with the dense network with a hand-crafted feature input. This is especially the case for predicting the pedestrians' intents earlier and further away from the crosswalk. Both the LSTM and the convolutional layers did not lead to the expected improvement. Especially the LSTM suffers from missing clues for significant movement changes in the pedestrians' trajectory. E.g. a head-tracking based on high resolution images could be helpful in this situation.

The evaluation of the temporal prediction horizon showed a very good accuracy for the investigated crossing pedestrians even for large times. For the given dataset the accuracy of the proposed dense neural network never dropped below 80% for the given time horizon of 6 s.

References

[1] C. Braeuchle, J. Ruenz, F. Flehmig, W. Rosenstiel, and T. Kropf, "Situation analysis and decision making for active pedestrian protection using bayesian networks," in *Proc. of the 6. Tagung Fahrerassistenz, München*, 2013.

[2] B. Völz, H. Mielenz, G. Agamennoni, and R. Siegwart, "Feature relevance estimation for learning pedestrian behavior at crosswalks," in *IEEE Int. Conf. on Intelligent Transportation Systems (ITSC)*, 2015.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[4] Z. Taylor, J. Nieto, and D. Johnson, "Multi-modal sensor calibration using a gradient orientation measure," *Journal of Field Robotics*, vol. 32, pp. 675–695, 2015.

[5] C. G. Keller and D. M. Gavrila, "Will the pedestrian cross? a study on pedestrian path prediction," *IEEE Trans. Intell. Transp. Syst.*, 2013.

[6] J. Kooij, N. Schneider, F. Flohr, and D. Gavrila, "Context-based pedestrian path prediction," in *Proc. of the European Conference on Computer Vision (ECCV)*, 2014.

[7] S. Schmidt and B. Färber, "Pedestrians at the kerb - recognising the action intentions of humans," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 12, pp. 300–310, 2009.

[8] S. Köhler, M. Goldhammer, S. Bauer, S. Zecha, K. Doll, U. Brunsmann, and K. Dietmayer, "Stationary detection of the pedestrian's intention at intersections," *IEEE Intell. Transp. Syst. Mag.*, 2013.

[9] R. Quintero, J. Almeida, D. F. Llorca, and M. A. Sotelo, "Pedestrian path prediction using body language traits," in *Proc. of the IEEE Intelligent Vehicles Symposium (IV)*, 2014.

[10] D. Ellis, E. Sommerlade, and I. Ried, "Modelling pedestrian trajectory patterns with gaussian processes," in *IEEE 12th International Conference on Computer Vision (ICCV) Workshops*, 2009.

[11] V. Karasev, A. Ayvaci, B. Heisele, and S. Soatta, "Intent-aware long-term prediction of pedestrian motion," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2016.

[12] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.

[13] 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*, vol. 15, pp. 1929–1958, 2014.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, 2012, pp. 1106–1114. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks

[15] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, "Theano: new features and speed improvements," Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[16] S. Dieleman, J. Schlter, C. Raffel, E. Olson, S. K. Snderby, D. Nouri, D. Maturana, M. Thoma, E. Battenberg, J. Kelly, J. D. Fauw, M. Heilman, diogo149, B. McFee, H. Weideman, takacsg84, peterderivaz, Jon, instagibbs, D. K. Rasul, CongLiu, Britefury, and J. Degrave, "Lasagne: First release." Aug. 2015. [Online]. Available: http://dx.doi.org/10.5281/zenodo.27878

[17] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," *CoRR*, vol. abs/1212.5701, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[18] A. Teichmann, J. Levinson, and S. Thrun, "Towards 3d object recognition via classification of arbitrary object tracks," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.