FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

ROBUST MACHINE LEARNING AND THE APPLICATION TO LANE CHANGE DECISION MAKING PREDICTION

By

HUA HUANG

A Dissertation submitted to the
Department of Mathematics
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

2021

Hua Huang defended this dissertation on March 26, 2021.
The members of the supervisory committee were:

Adrian Barbu

Professor Directing Thesis

Xiuwen Liu

University Representative

Kyle Gallivan

Chair of the Examination Committee

Giray Okten

Committee Member

Mark Sussman

Committee Member

The Graduate School has verified and approved the above-named committee members, and certifies that the dissertation has been approved in accordance with university requirements.

Dedicated to my parents & sister

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Adrian Barbu, for his guidance on this thesis. Without his inspirational guidance, encouragement and support, it will be impossible for me to complete this dissertation. I also want to thank my entire dissertation committee for their valuable insights and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

In the foreseeable future, autonomous vehicles will have to drive alongside human drivers. In the absence of vehicle-to-vehicle communication, they will have to be able to predict the other road users' intentions. Equally importantly, they will also need to behave like a typical human driver such that other road users can infer their actions. It is critical to be able to learn a human driver's mental model and integrate it into the planning and control algorithm. In this thesis, we first present a robust method to predict lane changes as cooperative or adversarial. For that, we first introduce a method to annotate lane changes as cooperative and adversarial based on the entire lane change trajectory. We then propose to train a specially designed neural network to predict the lane change label before the lane change has occurred and quantify the prediction uncertainty. The model will make lane change decisions following human drivers' driving habits and preferences, i.e., it will only change lanes when the surrounding traffic is considered to be appropriate for the majority of human drivers. It will also recognize unseen novel samples and output low prediction confidence correspondingly, to alert the driver to take control in such cases.

CHAPTER 1

INTRODUCTION

1.1 Problem Formalization

Deep learning has many successful applications in object classification, language translation, game playing and many other tasks [LeCun et al., 2015]. By stacking multiple simple and non-linear feed-forward layers of neurons, deep learning can learn the representation automatically. Based on the Universal Learning Theory [Hornik, 1991], one single layer with sufficiently many hidden units can approximate a function over a compact input set to arbitrary accuracy.

Even though deep learning is a very powerful tool, it can be fooled easily by adversarial samples [Goodfellow et al., 2014b, Nguyen et al., 2015]. In many cases, neural networks are overconfident in their predictions [Guo et al., 2017]. Neural networks based on the de facto activation function, ReLU [Nair and Hinton, 2010], which is defined as

$$f(x) = \max(x, 0), \tag{1.1}$$

have been proved to produce almost always high confidence predictions far away from the training data [Hein et al., 2019].

Overconfidence on unseen novel samples is highly undesirable for safety-critical applications. In the first fatality caused by an autonomous driving system, the perception system confidently predicted the white side of a trailer as bright sky [Kendall and Gal, 2017] and the autonomous system did not signal any warning or engage any brake. In a more recent fatality caused by an autonomous driving system, the perception system classified a pedestrian with a bicycle consequently as vehicle, other, vehicle, bicycle, other and changed the label multiple times until 1.2 seconds before the collision [NTSB, 2019]. Another application in which overconfidence is highly undesirable is medical image diagnosis, e.g., for novel or rare diseases, the machine learning system should be able to recognize the novelty and alert a human operator [Schulam and Saria, 2015].

In comparison to the machine learning models learning from training datasets, human beings acquire skills and knowledge through learning from previous experience. Different from machine

learning models, human beings have the capability to recognize their own ignorance [Amodei et al., 2016], i.e., know when they don't know. If one wants to apply neural networks to safety-critical domains like autonomous driving and medical image diagnosing, the model has to be able to assess its prediction confidence and detect out-of-distribution (OOD) novel samples, i.e., know when it doesn't know.

1.2 Overview

The remaining of the thesis is organized as follows.

In chapter II, robust machine learning is formalized mathematically. Uncertainties in machine learning prediction are briefly discussed. A review of the literature on uncertainty quantification and OOD sample detection will be given. A version of the Compact Support Neuron Network (CSNN) is proposed.

Lane change decision making in an interactive environment is presented in Chapter III. A literature review of lane change decision making models in an interactive environment is given at the beginning. The extraction of lane changes in a naturalistic human driving dataset is then presented. An annotation method that can extract human drivers' preferences and driving habits in lane changes is shown at the end of the chapter.

Experiments with reliable neural networks are provided in Chapter IV. The radius penalty effect on the compactness of support is investigated first on the Moons dataset. The in-distribution prediction accuracy and OOD detection capability of CSNN with radius penalty is demonstrated on the NGSIM dataset. A similar test accuracy for in-distribution samples compared with normal neuron based networks, and a very high Area under the Receiver Operating Characteristic curve (AUROC) are obtained with the CSNN algorithm. The performance of CSNN is also compared with recently developed OOD detection algorithms.

The conclusion will be given in Chapter V.

CHAPTER 2

ROBUST MACHINE LEARNING

For safety-critical applications, machine learning models should not only have high accuracy in predicting in-distribution samples, but also should be able to recognize unseen novel OOD samples. When the prediction confidence is small and uncertainty is high, a conservative action should be taken and the model can also alert a human to take over. In this thesis, a robust machine learning model will be developed.

Only recently did the machine learning community pay attention to the calibration of neural network based machine learning models [Kuleshov and Liang, 2015, Guo et al., 2017]. The probability of an event predicted by a well-calibrated model should reflect the true frequency, e.g., if the model predicts that 90% probability the lag vehicle will respect the merge request, then 90% of the time, the lag vehicle will respect the merge request. Different from traditional neural networks, modern neural networks are found to be no longer well calibrated [Guo et al., 2017, Nguyen et al., 2015].

2.1 Robust Machine Learning

Assume there is a data generation distribution $p(y, \boldsymbol{x})$ over the in- and out-of-distribution data, where labels $y \in \{1, ..., K\}$, K is the total number of classes, i.e., we focus on classification problems, and $\boldsymbol{x} \in R^d$, where d denotes the input feature size. The neural network-based model $p_\theta(y|\boldsymbol{x})$ is trained on the in-distribution training dataset $\mathcal{X}_{ind}$ and learns the in-distribution mapping $p(y|\boldsymbol{x}, \boldsymbol{x} \in \mathcal{X}_{ind})$, where θ are the parameters in the network.

After deploying the trained model to the real world, the sample fed in can come from both in-distribution $\mathcal{X}_{ind}$ and out-of-distribution $\mathcal{X}_{ood}$,

$$p(y|\boldsymbol{x}) = p(y|\boldsymbol{x}, \boldsymbol{x} \in \mathcal{X}_{ind})p(\boldsymbol{x} \in \mathcal{X}_{ind}) + p(y|\boldsymbol{x}, \boldsymbol{x} \in \mathcal{X}_{ood})p(\boldsymbol{x} \in \mathcal{X}_{ood}) \ . \tag{2.1}$$

We only expect the trained model to generalize well in $\mathcal{X}_{ind}$ and we simply do not know what will happen for OOD samples, hence for OOD samples, we want an equal confidence for all the

possible outcomes, i.e.,

$$p(y|\boldsymbol{x}, \boldsymbol{x} \in \mathcal{X}_{ood}) = \frac{1}{K} \ . \tag{2.2}$$

The uniform distribution has the maximum entropy. Setting the OOD prediction to be uniform also minimizes the Minimax Uncertainty Risk [Liu et al., 2020a].

Robust machine learning is defined as a model that can quantify $p(\boldsymbol{x} \in \mathcal{X}_{ind})$, i.e. is capable of separating in-distribution and OOD samples.

2.2 Uncertainties in Machine Learning Prediction

Prediction uncertainty of machine learning can be classified into two categories: epistemic (model) uncertainty and aleatoric uncertainty [Kendall and Gal, 2017].

Epistemic uncertainty measures the uncertainty in training the model parameters given the training data. Epistemic uncertainty is reducible as the size of training data increases. Averaging the predictions using ensembles has also been proved to be successful in reducing model uncertainties [Lakshminarayanan et al., 2017, Snoek et al., 2019].

Aleatoric uncertainty describes the noise inherent in the data. Aleatoric noise can be further decomposed into data uncertainty and distributional uncertainty. Data uncertainty is the irreducible uncertainty that arises from the natural complexity of the data, such as class overlap, label noise, homoscedastic and heteroscedastic noise. Homoscedastic noise does not change for different samples, while heteroscedastic noise depends on the samples inputted, with some samples having more noise than others. A detailed example is the human's driving behavior in lane changes. Drivers' behaviors and preferences in lane change are inherently highly uncertain. When a human driver intends to initialize a lane change but the window is tight, a timid driver or a patient driver may chooses to wait for a more comfortable window, while an aggressive or impatient driver will choose to turn immediately. When there is a moderate or large window, human drivers will tend to turn immediately, which leads to low uncertainty. In addition to a broad spectrum of personal preferences and driving habits, a single human driver's behavior is often peculiar and irrational [Driggs-Campbell et al., 2017] and generally does not satisfy the Markov property. Simply saying, there is significant overlap between classes.

Distributional uncertainty arises due to the mismatch between training and test distributions. When a predictor is fed samples far from the distribution of its training dataset, the credibility of

the prediction should be low. The novelty of out-of-distribution (OOD) samples in lane changes can come from the fact that the training dataset is collected at a single site, while different cities have different driving habits. Even for the same site, at a different time of the day such as rush hour vs. non-rush hour, in different weather conditions such as sunny and rainy days, human drivers will also behave differently. This uncertainty can be minimized by training the model on multiple distributions but cannot be eliminated completely. There will always be long-tail unlikely events that the algorithm cannot anticipate and they are so rare that they almost never happen. It is crucial to be able to detect these OOD samples if a data-driven approach is adopted in the Planning and Control algorithm of an Autonomous Vehicle (AV).

A machine learning model that does not have the capability of detecting the similarity between a testing sample and training samples will lead to unpredictable and even disastrous results. A detailed example is the recent collision between a self-driving vehicle with a pedestrian [NTSB, 2019], a direct quote from the highway accident report provided by National Transportation Safety Board says:

"The system never classified her as a pedestrian — or correctly predicted her path—because she was crossing N. Mill Avenue at a location without a crosswalk, and the system design did not include consideration for jaywalking pedestrians. The ADS changed the pedestrian's classification several times, alternating between vehicle, bicycle, and other. Because the system never classified the pedestrian as such, and the system's design excluded tracking history for nonpersisting objects — those with changed classifications — it was unable to correctly predict the pedestrian's path."

The perception systems backed by machine learning are only able to classify an object as pedestrian if and only if the object is near a crosswalk, while a pedestrian can happen to cross a street with a crosswalk or without a crosswalk in the real world. Incapability to include the whole scenarios in the training and failure to detect the novelty contribute to the disaster. In this thesis, we focus on quantifying distributional uncertainty and OOD sample detection.

2.3 Uncertainty Quantification and out-of-distribution Sample Detection

Uncertainty quantification and OOD detection algorithms can be categorized into Bayesian and non-Bayesian based approaches.

2.3.1 Bayesian Approaches

Bayesian neural networks [Kendall and Gal, 2017, Depeweg et al., 2018, Sun et al., 2018, Hafner et al., 2020, Gustafsson et al., 2020] combine the predictive power of a neural network with Bayesian uncertainty quantification. A prior $q(\theta)$ is designed to induce a distribution over functions that the network actually approximates. It is a challenge to specify the prior and it's still an open question how to do it. The epistemic uncertainty is obtained through Bayesian inference

$$p(y|\boldsymbol{x}, \mathcal{D}) = \int p(y|\boldsymbol{x}, \theta) p(\theta|\mathcal{D}) d\theta \ , \tag{2.3}$$

in which $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$ is the finite training dataset and n is the total number of samples. Uncertainties in the network parameters θ are marginalized out to obtain the posterior distribution. The true posterior $p(\theta|\mathcal{D})$ is typically approximated with a prior $q(\theta)$. The integral is intractable and methods like Monte Carlo are utilized to approximate it,

$$p(y|\boldsymbol{x}, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} p(y|\boldsymbol{x}, \theta^i), \quad \theta^i \sim q(\theta) \ , \tag{2.4}$$

in which N is the number of Monte Carlo simulations. Variational Inference[Graves, 2011, Sun et al., 2018, Blundell et al., 2015, Gustafsson et al., 2020, Gal and Ghahramani, 2016] is the principled approach to build the $q(\theta)$, in which a variational parameter ϕ is adopted to construct a distribution $q_\phi(\theta)$. The parameter ϕ is determined by minimizing the Kullback-Leibler (KL) divergence with respect to the data posterior $p(\theta|\mathcal{D})$.

The Prior Network [Malinin and Gales, 2018] proposes to explicitly model distributional uncertainty by adding a prior distribution over predictive distributions. Eq. 2.3 is modified as

$$p(y|\boldsymbol{x}, \mathcal{D}) = \int \int p(y|\boldsymbol{\mu}) p(\mu|\boldsymbol{x}, \theta) p(\theta|\mathcal{D}) d\boldsymbol{\mu} d\theta \ , \tag{2.5}$$

in which $\boldsymbol{\mu}$ is a categorical distribution over class labels, i.e.,

$$\boldsymbol{\mu} = [p(y = y_1), \ p(y = y_2), \ ..., \ p(y = y_K)]^T \ . \tag{2.6}$$

The terms in the integral are data, distributional, and model uncertainty. Prior Network is a hierarchical model as model uncertainty will affect distributional uncertainty, and distributional uncertainty will affect data uncertainty. When model uncertainty is marginalized out,

$$p(y|\boldsymbol{x}, \mathcal{D}) = \int p(y|\boldsymbol{\mu}) [\int p(\boldsymbol{\mu}|\boldsymbol{x}, \theta) p(\theta|\mathcal{D}) d\theta] d\boldsymbol{\mu} = \int p(y|\boldsymbol{\mu}) p(\boldsymbol{\mu}|\boldsymbol{x}, \mathcal{D}) d\boldsymbol{\mu} \ , \tag{2.7}$$

it will give the data and distributional uncertainty given model uncertainty. The marginalization is intractable and needs to be approximated.

Certified Certain Uncertainty (CCU) [Meinke and Hein, 2019] models in-distribution and OOD samples separately in Bayesian frameworks. The posterior $p(\boldsymbol{x}|i)$ and $p(\boldsymbol{x}|o)$, where i stands for in-distribution and o indicates out-of-distribution, are modeled with generative Gaussian mixture models (GMM) as

$$p(\boldsymbol{x}|i) = \sum_{k=0}^{K_i} \alpha_k \ \exp(-\frac{d(\boldsymbol{x}, \boldsymbol{\mu}_k)^2}{2\sigma_k^2}) \ , \tag{2.8}$$

$$p(\boldsymbol{x}|o) = \sum_{l=0}^{K_o} \beta_l \ \exp(-\frac{d(\boldsymbol{x}, \boldsymbol{v}_l)^2}{2\theta_l^2}) \ , \tag{2.9}$$

where K_i and K_o are the numbers of centroids and the modified distance is defined as

$$d(\boldsymbol{x}, \boldsymbol{y}) = ||C^{-\frac{1}{2}}(\boldsymbol{x} - \boldsymbol{y})||_2 \ , \tag{2.10}$$

where C denotes a modified covariance matrix of the in-distribution data, weights α_k and β_l are defined as

$$\alpha_k = \frac{1}{K_i} \frac{1}{(2\pi\sigma_k^2 \det(C))^{\frac{d}{2}}} \ , \tag{2.11}$$

$$\beta_l = \frac{1}{K_o} \frac{1}{(2\pi\theta_l^2 \det(C))^{\frac{d}{2}}} \ . \tag{2.12}$$

Centroids $\boldsymbol{\mu}_k$, $\boldsymbol{v}_l$ and variances σ_k^2, θ_l^2 are modeled with Maximum Likelihood Estimation (MLE). Due to the posterior $p(\boldsymbol{x}|i)$ and $p(\boldsymbol{x}|o)$ being modeled as Gaussian Radial Basis Functions (RBF), CCU can be proved to produce a uniform prediction far away from the training data.

Noise Contrastive Priors (NCP) [Hafner et al., 2020] adds input prior and output prior to train the model to output high uncertainty for OOD samples in regression tasks. The data prior is of the form

$$p_{prior}(\boldsymbol{x}, y) = p_{prior}(\boldsymbol{x}) p_{prior}(y|\boldsymbol{x}) \ , \tag{2.13}$$

where $p_{prior}(\boldsymbol{x})$ indicates the input prior and $p_{prior}(y|\boldsymbol{x})$ denotes the output prior. The OOD samples are perturbed inputs drawn from the input prior. The loss function includes a typical cross entropy loss and a KL-divergence between the output prior and training distribution over OOD inputs.

2.3.2 Non-Bayesian Approaches

For modern deep neural networks with many layers and up to hundreds of millions of weights, it is difficult to specify an appropriate model prior. It is also expensive to carry out a large number of Monte Carlo simulations for a large neural network. Non-Bayesian approaches are conceptually less complicated and computationally less expensive.

There are primarily five types of OOD detection techniques, i.e., deep ensemble, incorporating OOD samples in training, modifying classifier, adding a reject option, and variations of Radial-Basis-Function (RBF) networks.

Deep ensembles [Lakshminarayanan et al., 2017] have been proved to work well in high dimensional applications as individual networks tend to disagree on OOD samples with each other, which eventually leads to a higher prediction entropy. The entropy of the average prediction is used as the score for OOD detection.

$$p(y|\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} p_{\theta_i}(y|\boldsymbol{x}) \;, \tag{2.14}$$

$$H(p(y|\boldsymbol{x})) = -\sum_{i=0}^{K} p(y_i|\boldsymbol{x}) \; \log p(y_i|\boldsymbol{x}) \;, \tag{2.15}$$

where $\{\theta_i\}_{i=1}^{N}$ are the parameters for each individual network, and N denotes the number of ensemble networks. Deep ensembles can also be viewed as approximate Bayesian inference engines, in which the point estimates $\{\theta_i\}_{i=1}^{N}$ can be viewed as samples from the distribution $q(\theta)$, which in turn approximates the posterior $p(\theta|\mathcal{D})$. Deep ensembles have been found to outperform the popular Bayesian approach Monte Carlo (MC)-dropout [Gal and Ghahramani, 2016] in a recent study [Gustafsson et al., 2020]. The authors argued that due to the random initialization of the weights in each individual network, deep ensembles can capture multi-modality in the posterior distribution $p(\theta|\mathcal{D})$.

The second approach is to modify the training process by incorporating OOD samples and a hybrid loss function will be minimized to penalize the high confidence predictions on OOD samples [Hein et al., 2019, Liang et al., 2018, Lee et al., 2018b, Ren et al., 2019, Hendrycks et al., 2018, Meinke and Hein, 2019]. For example, Lee et al. proposed to minimize a modified Generative

Adversarial Networks (GAN) loss [Lee et al., 2018b]:

$$\begin{aligned} \min_G \max_D \quad & \beta \quad \boldsymbol{E}_{P_G(\boldsymbol{x})}[KL(u(y)||P_\theta(y|\boldsymbol{x}))] \\ & + \quad \boldsymbol{E}_{P_{in}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \boldsymbol{E}_{P_G(\boldsymbol{x})}[\log(1 - D(\boldsymbol{x}))] \ , \end{aligned} \tag{2.16}$$

in which G is the generator, D indicates the discriminator, β is the Kullback–Leibler (KL) divergence loss coefficient, and u is the uniform categorical distribution. The KL divergence is defined as

$$KL(Q||P) = \sum_{\boldsymbol{x} \in X} Q(\boldsymbol{x}) \log(\frac{Q(\boldsymbol{x})}{P(\boldsymbol{x})}) \ . \tag{2.17}$$

Minimization of the KL divergence between model output and the uniform distribution will force the model to output uniform prediction in the generated samples. The second and third terms form the original GAN [Goodfellow et al., 2014a] loss. The major disadvantage of introducing OOD samples in training is that the space of OOD samples will be too large to cover. The model trained on one set of OOD samples might not be able to detect another unseen set of OOD samples. In another recent work found that under this setting, the generated OOD samples are only around part of the boundaries of in-distribution data, i.e., it does not cover the entire boundary or the whole OOD domain, hence it still outputs arbitrarily high confidences for OOD samples far from the in-distribution samples [Vernekar et al., 2019].

The third type is classifier-based. An OOD sample is argued to have a lower maximum class probability $p(\hat{y}|\boldsymbol{x}) = \max_k p(y = k|\boldsymbol{x})$ than an in-distribution sample, hence the maximum class probability can be compared to detect OOD samples [Hendrycks and Gimpel, 2017, Liang et al., 2018]. Temperature scaling has also been introduced to further separate the in-distribution samples from OOD samples [Liang et al., 2018, Hsu et al., 2020].

$$S_i(\boldsymbol{x}; T) = \frac{\exp(f_i(\boldsymbol{x})/T)}{\sum_{j=1}^{K} \exp(f_j(\boldsymbol{x})/T)} \ , \tag{2.18}$$

in which T is the temperature scaling parameter and f is the logit. Energy scores [Liu et al., 2020b] have also been found to better separate the in-distribution samples from OOD samples compared with the softmax scores,

$$s = -\log(\sum_{j=1}^{K} e^{f_j(\boldsymbol{x})}) \ . \tag{2.19}$$

Classifiers have also been combined with nearest neighbours to define the trustworthiness of predictions [Jiang et al., 2018]. The metric of the trust score is defined as the ratio between the distance from a testing sample to the nearest class different from the predicted class and the distance to the predicted class. Similarly, the confidence score [Lee et al., 2018a] has also been defined based on the Mahalanobis distance between the testing sample and the class conditional Gaussian distribution, i.e.,

$$s = \max_c -(\mathbf{f}(\boldsymbol{x}) - \widehat{\boldsymbol{\mu}_c})^T \widehat{\Sigma}^{-1} (\mathbf{f}(\boldsymbol{x}) - \widehat{\boldsymbol{\mu}_c}) \ , \tag{2.20}$$

where c is the class, $\mathbf{f}(\boldsymbol{x})$ denotes the output of the penultimate layer of neural network, and the class mean $\widehat{\boldsymbol{\mu}_c}$ is defined as

$$\widehat{\boldsymbol{\mu}_c} = \frac{1}{N_c} \sum_{i:y_i=c} \mathbf{f}(\boldsymbol{x}_i) \ , \tag{2.21}$$

where N_c is the total number of samples belong to class c. The class covariance $\widehat{\Sigma}$ is defined as

$$\widehat{\Sigma} = \frac{1}{N} \sum_c \sum_{i:y_i=c} (\mathbf{f}(\boldsymbol{x}_i) - \widehat{\boldsymbol{\mu}_c})(\mathbf{f}(\boldsymbol{x}_i) - \widehat{\boldsymbol{\mu}_c})^T \ . \tag{2.22}$$

The fourth type is adding a reject option to the model output [Cortes et al., 2016, Vernekar et al., 2019, Mohseni et al., 2020, Geifman and El-Yaniv, 2017, 2019]. OOD detection can be formalized as learning with rejection. For example, in a bird species classification task given a set of pictures, pictures of birds need to be correctly predicted, but pictures of non-bird need to be rejected. A classifier and a reject function are jointly learned in these settings. Selective classification [Tax and Duin, 2008] is another formalization in which class densities are estimated and a testing sample is assigned to the class with the largest posterior probability.

Lastly, compact support networks have also been introduced through variations of Radial-Basis-Function (RBF) networks [Van Amersfoort et al., 2020, Liu et al., 2020a]. DUQ [Van Amersfoort et al., 2020] computes a feature vector with a Multi-Layer Perceptron (MLP) and then calculates the distance between the feature vector and the class centroids $\boldsymbol{e}_c$. The class centroid is updated with an exponential moving average of the feature vectors belonging to that class. When the distance between the feature vector and any class centroid is large, it is considered to be an OOD sample. A hybrid loss function of gradient penalty and logistic loss is minimized,

$$\ell = \lambda[|| \bigtriangledown_{\boldsymbol{x}} \sum_c K_c||_2^2 - 1]^2 - \sum_c [y_c \log(K_c) + (1 - y_c) \log(1 - K_c)] \ , \tag{2.23}$$

in which the prediction uncertainty is defined as

$$K_c = \exp[-\frac{\frac{1}{n}||W_c f_\theta(\boldsymbol{x}) - \boldsymbol{e}_c||_2^2}{2\sigma^2}] \ . \tag{2.24}$$

A relatively new large-scale benchmark study of recently proposed methods on uncertainty quantification and OOD detection is given in Ovadia et al. [2019]. The authors found that among all the newly proposed algorithms, deep ensemble achieved the best performance in many metrics and an ensemble of small size, e.g., 5 networks might be enough to achieve a good performance.

2.4 The Compact Support Neural Network

The Compact Support Neural Network (CSNN) [Barbu and Mou, 2021] has the same-level of accuracy in predicting in-distribution samples compared with a normal neuron-based network. It will have zero output for samples outside the support, i.e., OOD samples. In this thesis, CSNN will be adopted to predict human drivers' lane change decision makings and detect unseen OOD samples.

CSNN smoothly combines the ReLU-type activation [Nair and Hinton, 2010] with a traditional Radial Basis Function (RBF) network [Broomhead and Lowe, 1988] through a shape parameter α. A RBF function is a function defined as

$$\phi_{\boldsymbol{c}}(\boldsymbol{x}) = \phi(||\boldsymbol{x} - \boldsymbol{c}||) \ , \tag{2.25}$$

in which $\boldsymbol{c}$ is the kernel center and the distance is typically Euclidean distance. The most popular RBF is the Gaussian RBF, which is defined as

$$\phi(r) = e^{-(\epsilon r)^2} \ , \tag{2.26}$$

where ϵ is a shape parameter. The ReLU activation function is defined as

$$f(x) = \max(0, \ x) \ . \tag{2.27}$$

The neuron with compact support in CSNN is defined as

$$\begin{aligned} f_{\boldsymbol{\mu}}(\boldsymbol{x}) &= \max(R^2 - ||\boldsymbol{x} - \boldsymbol{u}||^2, 0) \\ &= \max(\alpha(R^2 - \boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{u}^T\boldsymbol{u}) + 2\boldsymbol{u}^T\boldsymbol{x}, 0) \ . \end{aligned} \tag{2.28}$$

When the shape parameter $\alpha = 0$, it will be a standard ReLU neuron. When $\alpha > 0$, it will be a Compact Support Neuron. In practice, CSNN is trained starting from a normal neuron-based network ($\alpha = 0$) and then gradually increasing the shape parameter α to 1. It can be shown that the neuron only has support within a sphere of radius

$$R_\alpha^2 = R^2 + ||\boldsymbol{u}||^2(\frac{1}{\alpha^2} - 1) \tag{2.29}$$

and center

$$\boldsymbol{c} = \frac{\boldsymbol{u}}{\alpha} , \tag{2.30}$$

where R and $\boldsymbol{u}$ are learnable parameters. The neurons with compact support are used in the penultimate layer of the neural network. The last layer of the network will be a fully connected layer with weights $\mathbf{g}$ and the logits are computed as

$$logit(\boldsymbol{x}) = \mathbf{g} \cdot f_{\boldsymbol{\mu}}(\boldsymbol{x}) . \tag{2.31}$$

The probabilities are computed through the softmax function

$$p(y|\boldsymbol{x}) = \frac{e^{logit_y(\boldsymbol{x})}}{\sum_{i=1}^{K} e^{logit_i(\boldsymbol{x})}} . \tag{2.32}$$

The parameters are optimized by maximum likelihood estimation. To further constrain the support, radius penalties are added to the loss function. Experiments show that the infinity norm works best. For this binary classification task, the overall loss is the binary cross entropy loss plus the radius penalty,

$$\ell = \sum_i y_i \log(p_i) + (1 - y_i) \log(1 - p_i) + \lambda ||R_\alpha||_\infty , \tag{2.33}$$

where λ is the radius penalty coefficient.

It is worth noting that the ability to measure the distance from a testing sample to the training dataset is a necessary condition to get a high-quality estimation of distributional uncertainty [Liu et al., 2020a]. The neuron's output in the CSNN is determined by the distance of the input to the neuron's parameter vector, hence satisfies the necessary condition.

2.5 Conclusion

In this chapter, robust machine learning is defined as a machine learning model that can separate in-distribution samples from OOD samples. Uncertainties in machine learning prediction are classified into three categories, out of which the distributional uncertainty is arguably the most imperative uncertainty the autonomous vehicle has to address. Bayesian and non-Bayesian approaches in uncertainty quantification and OOD sample detection are briefly summarized. The CSNN model elegantly interpolates between the ReLU-type NN and a traditional RBF network by introducing a shape parameter, hence enjoys great flexibility.

CHAPTER 3

LANE CHANGE DECISION MAKING IN INTERACTIVE ENVIRONMENT

One of the biggest obstacles in deploying autonomous vehicles is the necessity for the autonomous vehicles to interact with human drivers, especially in scenarios that need cooperation, e.g., lane changes, unprotected left turns, roundabouts, unsignaled intersections, etc. Without vehicle-to-vehicle communication, the autonomous vehicle has to be able to recognize other road users' intentions, and equally importantly, it has to behave like a vehicle driven by a typical human driver such that other road users can anticipate its actions. To accelerate the integration of autonomous vehicles with human-driven vehicles, the human driver's mental model must be learned and incorporated in the Planning & Control (P&C) algorithm. In this thesis, we will focus on predicting human drivers' behaviors in lane changes with machine learning.

The lane change is considered to be one of the most challenging maneuvers even for human drivers. Around 18% of all accidents happen during the execution of a lane change, and most of them are rear-end collision [Scheel et al., 2018]. As can be seen in Fig. 3.1, ego, i.e., the vehicle that carries out the lane change, has to consider both the relative position and relative velocity of the surrounding 3 obstacles. In particular, ego should understand the intention of the lag vehicle in the target lane, which could be one of the following:

- The lag vehicle will respect the cut-in request. If necessary, it will decelerate to create enough space.
- The lag vehicle will ignore the request and might even accelerate to close the window to deter ego from merging.
- The lag vehicle will keep its speed and wait for the ego's next action, i.e., a wait-and-see mode.

If ego does not recognize the environment and recklessly changes the lane, the lag vehicle in the target lane might have to do a harsh brake or be forced to change lane. In the worst scenario, a collision might happen with the lag vehicle.

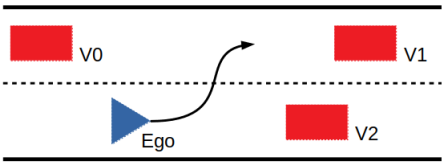

Figure 3.1: Lane change illustration.

Equally importantly, ego also has to behave like a typical human driver such that other road users can infer ego's intentions. For lane changes, ego can choose to:

- Carry out the lane change now.
- Decide that it is impolite or risky to carry out the lane change at the current time and abort the lane change.

Other road users will anticipate that ego will not carry out the lane change when the window is too tight and they will also cooperate with ego and let the ego cut in when there is reasonable space.

We assume we already have the intention to perform the lane change, and already know the target lane. The mission is to make a decision about whether to carry out the lane change now or wait until the next window. We focus on the impact of ego's lane change on the vehicle following in the target lane, i.e., to train ego to be a good citizen.

In addition to correctly classify human drivers' intentions, it is critical that the model be able to assess its prediction confidence. Safety is the No.1 priority in autonomous vehicles, and a simple decision in planning & control could be a mater of life and death. When the prediction confidence is low, the model should recommend a conservative action or alert a human to take over.

3.1 Lane Change Decision Making Models

In this section, a brief review of planning and control algorithms for lane change in interactive environments will be given.

3.1.1 Rule-Based Models

Rule-based models are the most discussed lane change decision models. Functions are proposed and the parameters of the function are typically calibrated with a naturalistic dataset. The Gipps model [Gipps, 1986] is a hierarchical lane change decision-making model, in which the deceleration

of the subject vehicle is used to evaluate the feasibility of the lane change. The Gipps model does not take the different strategies and behaviors of neighboring drivers into consideration. Based on the reaction of the lag vehicle, the ARTEMiS model [Hidas, 2005] classifies lane changes into free, forced, and cooperative types, and calls the corresponding algorithm for each scenario. The lane change model is based on a car-following model, and the feasibility of a lane change is determined by the gap, which in turn is decided by the acceptable acceleration/deceleration limits. MOBIL [Kesting et al., 2007] describes a method to decide whether to immediately perform a lane change or not. Safety is measured based on the braking deceleration imposed on the new follower in the target lane. If the overall benefit, defined as the comprehensive increase in accelerations of the ego and followers in the old and new lane, is larger than a threshold, the ego can carry out the lane change. A more thorough review of lane change models is available in Rahman et al. [2013].

3.1.2 Game Theory-Based Approaches

Lane change can also be modeled with game theory-based approaches. There are three major ingredients in a game, i.e., players, strategies, and payoffs (rewards). An agent's behavior is defined as cooperative if the joint utility is knowingly and willingly increased in comparison to a reference utility [Schwarting et al., 2018]. The interaction of through vehicles and merging vehicles has been modeled as a two-person non-zero-sum non-cooperative game of complete information [Kita, 1999]. The merging vehicle has strategies of merge and pass, and the through vehicle will either give way or not give way. The payoffs for either vehicle are based on the position and speed relative to the neighboring vehicles. The interactions between controlled vehicles and surrounding vehicles can be described through a collective cost function [Wang et al., 2015], where the controlled vehicles make decisions based on the expected behavior of other vehicles. In a cooperative lane change, the controlled vehicles coordinate their decisions to optimize the collective cost. Since the intentions of surrounding vehicles cannot be observed before carrying out a lane change, the lane change has also been described as a two-player game with incomplete information [Yu et al., 2018], in which the timing and acceleration of lane change are based on the degree of aggressiveness of the lag vehicle. The aggressiveness determines the driver's preferences of space payoff to safety payoff. A neural network-based model [Yan et al., 2018] has been utilized to model the payoffs and the authors found that the lag vehicle's strategy is more difficult to predict than the subject vehicle's.

3.1.3 Partially Observable Markov Decision Processes-Based Approaches

The lane changing maneuver is a typical time-sequential problem where the completion of the task involves a sequence of actions, and the performance of the current action has an impact on the ultimate goal of the task (i.e. a successful lane change). It is therefore suitable to be modeled as a Markov Decision Process (MDP) and can be solved with Reinforcement Learning (RL). Since the intentions of other road users are not directly observable, the problem needs to be formalized as a Partially Observable Markov Decision Process (POMDP), in which the intentions are hidden variables. POMDP problems are PSPACE complete [Schwarting et al., 2018], thus they are computationally intractable for high-dimensional applications. Most efforts in the literature have been devoted to solving it efficiently in an approximate manner, in which the state space is discretized into abstract concepts, e.g., whether a lane change is possible and beneficial. The actions are also abstract concepts, e.g., lane following, turn left/right, signal left/right, etc [Ulbrich and Maurer, 2015]. The discretization of state and action space needs lots of domain knowledge, and the hand-crafted representation can not be generalized to different situations. A continuous POMDP solver that can learn the representation of the specific situation automatically is proposed in Brechtel et al. [2014].

3.1.4 Deep Learning-Based Approaches

Encouraged by the successful applications of Deep Learning in object classification, language translation, game playing, and many other tasks [LeCun et al., 2015], there has been a surge of interest recently in applying Deep Learning to predict lane change decisions. A deep convolutional neural network has been utilized to classify whether it's safe to initialize a lane change [Jeong et al., 2017], in which the model is trained from images directly. The lane change task can be decomposed to decision making and implementation, which has been modeled with deep belief networks and long-short-term-memory (LSTM) correspondingly [Xie et al., 2019]. LSTM has also been adopted to model both the lane change and the lane following behaviors [Zhang et al., 2019]. Attention mechanisms have been introduced in Scheel et al. [2019] to improve the prediction accuracy.

Deep learning has also been applied to reinforcement learning to learn both the lane change decision making and the lane change implementation [Hoel et al., 2018, Chen et al., 2019, Alizadeh et al., 2019, Shi et al., 2019, Mirchevska et al., 2018]. The major disadvantage of reinforcement

learning is that a simulation bed has to be built, in which behaviors of agents need to be as close as possible to human drivers' behaviors. Compared with learning a good driving policy, characterizing high-fidelity agent driving behaviors is equally difficult. Another challenge is that the rewards need to be hand-crafted to be able to train a smooth and natural policy.

3.2 Lane Change Extraction

The FHWA's Next Generation Simulation (NGSIM) dataset [Kovvali et al., 2007] has been widely used to study human driving behaviors. The dataset contains videos of the northbound traffic on I-80 and southbound traffic on US-101. The detailed location and time are given in Table 3.1. The study site is approximately 500 m long for I-80 and 640 m long for US-101. As illustrated in Fig. 3.2, the vehicle positions were recorded every 0.1 s from video cameras mounted from the top of a 30-story building adjacent to the freeway in dataset I-80 and a 37-story building adjacent to the freeway in dataset US-101. The NGSIM dataset contains 11779 vehicle trajectories, out of which 11328 trajectories are carried out by cars.

Figure 3.2: Videotaping vehicles passing the study area.

Since motorcycles and trucks change lanes differently from cars, only lane changes carried out by cars are included in this research. Lane changes from/to the rightmost lane are also excluded as the rightmost lane is for ramp merging/diverging, in which vehicles have to finish the merging/diverging before the merge/diverge point, so drivers tend to behave differently than in a typical lane change. Lane changes without a lag vehicle on the target lane are discarded as we focus on the social impact of lane change on the other road users. When there are no obstacles in front of ego, either on the original lane or target lane, we put an imaginary obstacle $r_0 = 100\,m$ ahead of ego, in which $r_0 = 100\,m$ is the detection range of ego, and the velocity of the obstacle is the same as ego. Overall, 1558 lane changes are extracted from the I-80 dataset and 1290 lane changes from the US-101 dataset. The trajectories of 200 successful lane changes are plotted in Fig. 3.3.

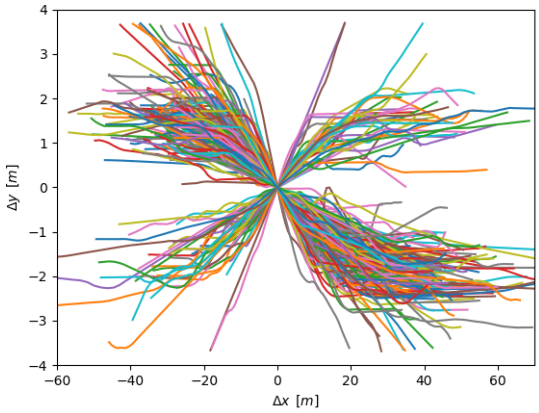

Figure 3.3: Lane changes.

3.3 Lane Change Annotation

To capture the likely actions of human drivers in lane changes and the likely reactions of the lag vehicle in the target lane, cooperative and adversarial lane changes will be defined and extracted from the data.

Since human drivers' intentions cannot be observed directly, there are various attempts in the literature to label cooperative/adversarial lane change behaviors. The positive (cooperative) samples are relatively easy to define, however, the negative (adversarial) samples are much more difficult to define. The fact that a human driver does not begin a lane change can be caused by

many factors, e.g., competing behavior of the lag vehicle in the target lane, he/she does not have an intention to carry out lane change and prefers car-following now. The implicit reasoning cannot be observed. For these reasons, different papers adopt different ways to label the negative samples:

- Negative labels for the lane change preparation stage. A lane change will experience a preparation process for seeking a suitable acceptable gap or adjusting the velocity before lane-changing execution, hence the lane-changing preparation stage is labeled as a number of lane-changing execution rejecting events [Scheel et al., 2018].
- Negative labels for observations before and after the ego starts the lateral move [Balal et al., 2016]. Car following maneuvers are also counted as negative samples. Under these conditions less than 0.1% samples are positive.
- Negative labels for decreases in relative longitudinal distance after lane change. The cooperative/adversarial strategy is labeled based on whether the relative distance between ego and lag vehicle in the target lane $\Delta x = x_{ego} - x_0$ decreases or not from time $t = -3\,s$ to $t = 0\,s$, in which $t = 0\,s$ is the time the ego crosses the lane divider. If the relative distance increases, it is labeled as cooperative [Yan et al., 2018].

Both the first and second method for labeling instances as negative are heuristic approaches and the extracted negative samples don't always fall in the adversarial category. For the third approach, the extracted lane changes from the I-80 dataset are labeled and plotted in Fig. 3.4. The plot shows Δx vs Δv at time $t = -3\,s$ for each trajectory. As one could see from Fig. 3.4, when the ego is slower than the lag vehicle in the target lane, i.e., $\Delta v = v_{ego} - v_0 < 0$, Δx will almost always decrease and the sample will be labeled as negative. This labeling will lead to overly conservative lane change decisions and cannot be used in moderate or heavy traffic.

Instead of looking at Δx, we inspect the deceleration of the lag vehicle in the target lane. Acceleration is the direct reflection of the impact lane change imposed on the other road users. It has also been used in decision-making models in Gipps [1986], Hidas [2005], Kesting et al. [2007]. If ego's lane change causes no forced harsh brake (a harsh brake is defined as a deceleration smaller than the comfortable deceleration $a_{comfortable} = -3\,m/s^2$ as recommended by the Institute of Transportation Engineers [Wolshon and Pande, 2016]) for the following vehicle, it will be labeled as a cooperative lane change. Since acceleration is calculated by the second-order derivative of the position and therefore can be noisy (see Fig. 3.5), we require the total duration of deceleration $a_t < -3\,m/s^2$ in $t \in [t_0,\ 5\,s]$ be less then 1 s for cooperative samples. Here $t_0 < 0$ is defined as the

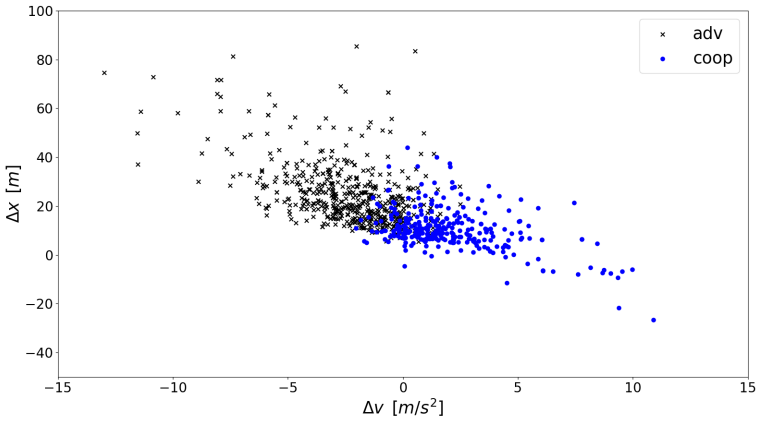

Figure 3.4: Label based on on the change in Δx.

time when the ego starts to have a lateral velocity $|v_y| > 0.213\,m/s$ [Scheel et al., 2018] and without oscillation thereafter. The time $t = 0\,s$ corresponds to when the ego crosses the lane divider. In such a way, the impact of ego's lane change on the lag vehicle is considered for the entire duration of the lane change.

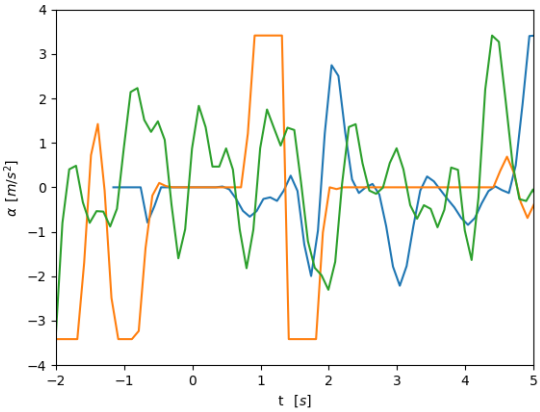

Figure 3.5: Accelerations of lag vehicles.

The other adversarial scenario is to abort the current window and merge after the current lag vehicle in the target lane. When it is deemed too aggressive or even dangerous to carry out lane change immediately, ego will prefer to wait for the next available window. In this scenario, we extract the lane changes and further require that at $t = -8\,s$ there is an open window, i.e., ego

Table 3.1: Statistics of the I-80 and US-101 Datasets.

	I-80	US-101
Location	Emeryville	Los Angeles
Time	4pm-4:15pm, 5pm-5:30pm	7:50am-8:35am
Samples	1558	1290
Merge in front coop	1095 (70.28%)	1116 (86.51%)
Merge in front adv	150 (9.63%)	32 (2.48%)
Merge after	313 (20.09%)	142 (11.01%)

chooses to abort the previous open window.

The statistics of the extracted lane change are summarized in Table 3.1. The extracted lane changes are plotted in Fig. 3.6 for the I-80 dataset and Fig. 3.7 for the US-101 dataset, in which the samples are scattered in the $\Delta v - \Delta x$ 2d space. As can be seen in Table 3.1, there are significantly more competing behaviors in the I-80 dataset. Two explanations for the differences in driving behaviors are:

- Traffic conditions. The traffic in US101 dataset is significantly less congested. The average velocity in the I-80 dataset is as low as 9.88 m/s, while the average velocity in the US-101 dataset is 12.54 m/s. In congested traffic, human drivers tend to be more aggressive in lane changes.
- Different cities have different driving habits.

More datasets like highD [Krajewski et al., 2018] need to be compared to further reveal the effects.

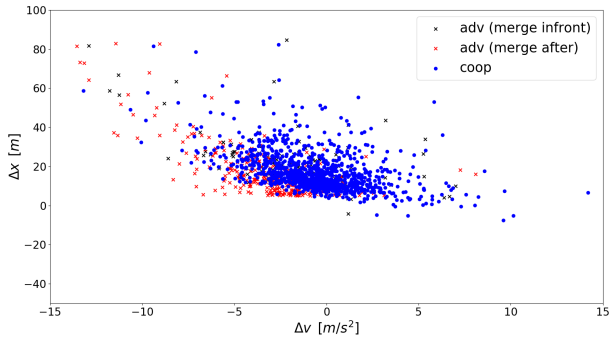

Figure 3.6: Lane changes in dataset I-80.

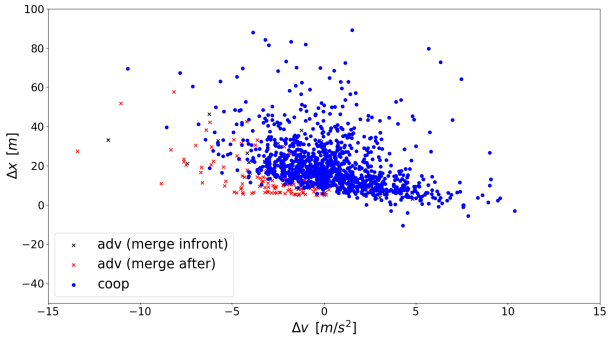

Figure 3.7: Lane changes in dataset US-101.

Another relevant observation is that there is class overlap, i.e., data uncertainty in both datasets. For one particular relative velocity, timid or patient drivers will choose to give up the current lane change window, while aggressive or impatient drivers will perform the lane change. Understandably, models should output low confidence in the ambiguous region, meanwhile, they are also required to have low confidence on OOD samples.

3.4 Conclusion

Without vehicle-to-vehicle communication, autonomous vehicles have to be able to predict other road users' intentions, they also have to act like typical human drivers, such that other road users can infer autonomous vehicles' actions. To accelerate the integration of autonomous vehicles with human driven vehicles, it is crucial to learn human drivers' mental model and incorporate it to the Planning & Control algorithm.

We briefly reviewed the planning and control algorithms for lane changes in interactive environments. The most investigated lane change decision-making model is rule-based, in which the acceleration of the subject vehicle is used to evaluate the feasibility of lane changes. Since the intentions of other road users cannot be observed directly, in game theory-based approaches, the game should be modeled as a game with incomplete information, and in POMDP-based approaches, the intentions are modeled as the hidden variables. Since lane change is a time-sequential problem, Recurrent Neural Networks have been adopted to model the lane change process. Deep reinforce-

ment learning is another promising approach as it can model both the lane change decision-making and the lane change implementation.

The lane changes are extracted from the naturalistic NGSIM dataset. We label the lane changes based on the deceleration of the lag vehicles in the target lane during the entire lane change process, i.e., train the autonomous vehicles to be good citizens. Window abortions are also labeled as negative samples. This way, human begins' driving behaviors and preferences in lane changes are extracted.

CHAPTER 4

EXPERIMENTS WITH COMPACT SUPPORT

In this chapter, we will begin with experiments on a toy dataset, i.e., the Moons dataset, during which we will show the learning mechanism of Compact Support Neurons and the impact of radius penalties. We will then apply the CSNN to the real-world NGSIM dataset, in which we will talk about the generation of OOD samples and give the in-distribution prediction accuracy and Area under the Receiver Operating Characteristic curve (AUROC) for OOD detection. We will also compare the performance of CSNN with other state-of-the-art algorithms. The generalization capability of CSNN is also investigated by training the algorithms on dataset I-80 and testing on dataset US-101, and vice versa.

4.1 Synthetic Dataset

To show the effectiveness of the CSNN in detecting OOD samples, CSNN models with α of 0 and 1 are trained on the moons dataset. The moons dataset contains two interleaving half circles corrupted by noise, one for each class, as illustrated in Fig. 4.1. We generated 2000 samples using the scikit-learn library [Pedregosa et al., 2011]. Another 10000 samples are generated on a uniform grid spanning $[-2.5,\ 3.5] \times [-3,\ 2]$. The samples are normalized to have 0 mean and standard deviation $1/\sqrt{d}$, in which d is the feature dimension, i.e., 2 here.

A two-layer CSNN model with 64 compact support neurons in the 1st layer is implemented. The output layer is a fully connected layer. The batch size is 64 and the learning rate is 0.001. The parameter R is initialized to 1 and is learnable. The radius penalty coefficient $\lambda = 0.64$. A MLP model is pretrained 50 epochs and the weights are used to initialize the CSNN models. The α is set to be 1 immediately after the pretraining for the model with $\alpha = 1$.

The samples and the prediction confidences are plotted in Fig. 4.1. With $\alpha = 0$, i.e., a ReLU-type neuron, the model will generalize the prediction far from the training dataset and output high confidence for OOD samples. With $\alpha = 1$, the confidence is 0.5 away from the two circles, while around the two circles, confidence is near 1. Just as designed, the compact support neuron

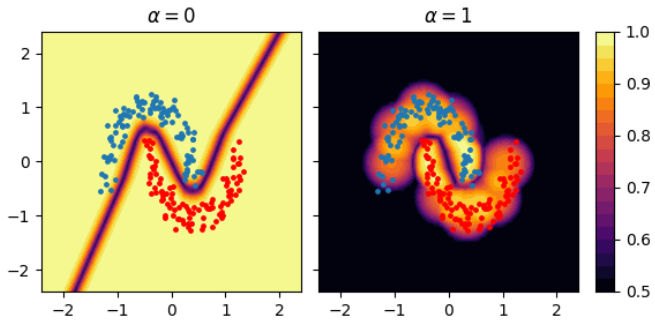

Figure 4.1: Confidence map for $\alpha = 0$ (left) and $\alpha = 1$ (right).

will shrink its support to a neighboring domain surrounding the training samples, correspondingly, CSNN will output low confidences for samples far from the training dataset. The confidence is also low for samples between the two moons, i.e., in addition to distributional uncertainty, the model successfully identifies the class overlap and data uncertainties [Malinin and Gales, 2018].

4.1.1 Radius Penalty Effect

The evolution in the input space of the support circles and the corresponding confidence contours with radius penalty $\lambda = 0$ and shape parameter $\alpha = 1$ are plotted in Fig. 4.2 and Fig. 4.3. The radius is initialized as 1 for the compact support neurons. As we can see in Fig. 4.2 and Fig. 4.3, there is still support encompassing the training samples and there are high confidences around the two moons, i.e., the support can be further constrained. To demonstrate the effectiveness of radius penalty, the radius penalty coefficient λ is gradually increased from 0 until 0.64. The corresponding support circles, confidence contour, and the largest and average radius after 500 epochs of training are given in Fig. 4.4.

As we can see in these figures, the radius penalty significantly changes the support circles and the confidence. Without radius penalty, i.e., $\lambda = 0.0$, the average radius of support circles decreases from 1 to 0.795, while the largest radius actually increases to 1.665. An increase in the radius for the largest support circle is also depicted in Fig. 4.2. Arguably the minimization of the cross entropy loss itself does not encourage the shrinkage of the support radius, while in detecting OOD samples, we want the compact circles to cover the in-distribution samples only, i.e., we want to have a support as tight as possible. When λ is increased to a small value $\lambda = 0.01$, the support circles are noticeably smaller as the largest radius decreases to 1.419. When λ is further increased

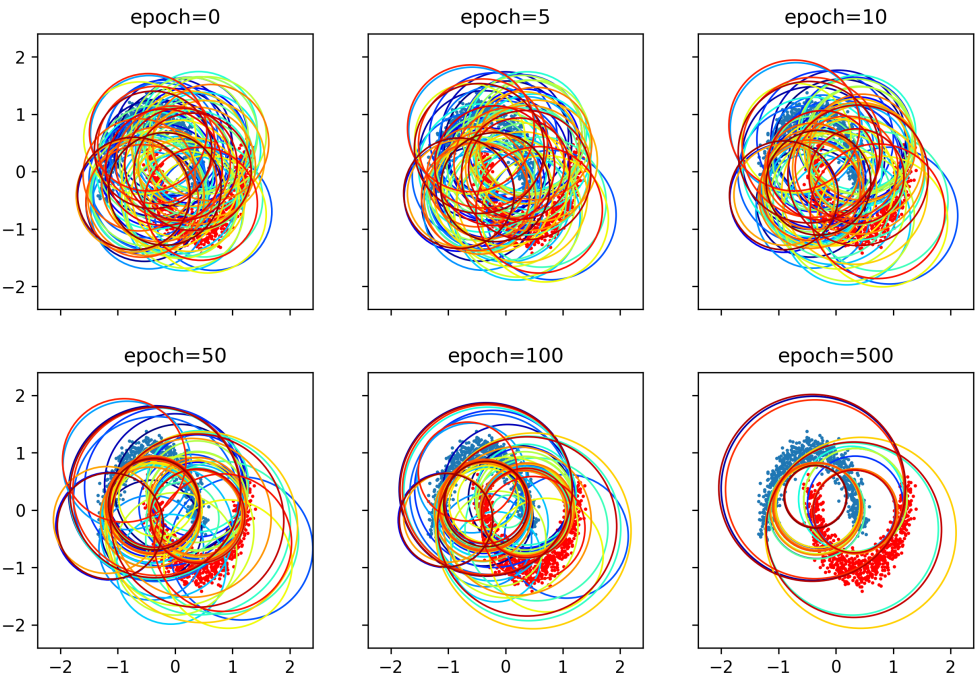

Figure 4.2: Evolution of the support circles during training.

to $\lambda = 0.04$, the group of support circles changes significantly, i.e., the two largest circles are broken up into 4 sets of large circles, which leads to undesirable large areas of support around the ends of moons. When λ is enlarged to $\lambda = 0.64$, again it significantly changes the group of circles as the largest radius converges to almost the same as the average radius, i.e., all the support circles have the same radius when the training has converged. The support circles cover the training samples nicely and the high confidence region tightly mimics the training samples. Once deviating from the training dataset, there is a sharp decrease in confidence.

4.1.2 OOD Detection

To increase the difficulty of OOD detection, we increase the noise level in generating the moons dataset to 0.18 such that there will be a significant portion of class overlap in the data. The OOD samples are generated on a uniform grid.

First, all the 2000 in-distribution samples are normalized to have 0 mean and 1 std in each dimension.

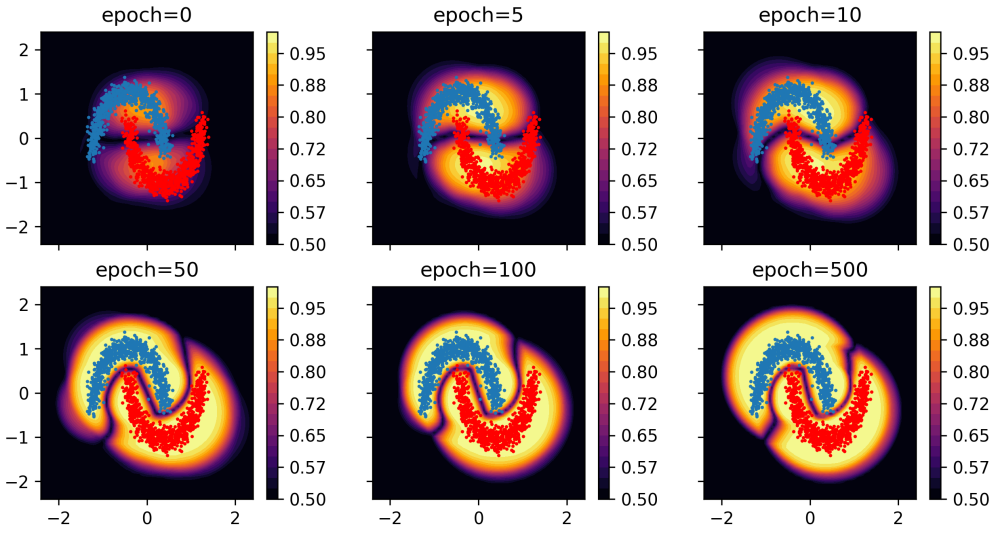

Figure 4.3: Evolution of the confidence during training.

Then for each in-distribution sample $\boldsymbol{x}_i$, the minimum distance d_i to other in-distribution samples is computed. Then we find a distance threshold τ as the 99 percentile of the d_i values, thus 99% of the d_i will be less than τ.

Then we generate 10000 samples on a uniform grid in the hyper-rectangle

$$\mathcal{R} = [-2.5, 3.5] \times [-3, 3] \tag{4.1}$$

The generated samples are then transformed using the mean and std of the in-distribution samples and the minimum distance to any in-distribution samples is calculated. The generated samples with a distance $d_i < \tau$ are discarded. This way, 7813 generated samples are kept as OOD samples. In comparison, there are 2000 in-distribution samples. The generated samples are plotted in Fig. 4.5. 75% of in-distribution samples are used for training and the remaining samples for test. The shape parameter $\alpha = 1$ and the radius penalty coefficient $\lambda = 0.64$.

The AUROC score for classifying between in-distribution and OOD samples will be used to gauge the OOD detection performance. AUROC is a threshold independent metric. The Receiver Operating Characteristic (ROC) curve plots the true positive rate TPR (also called sensitivity)

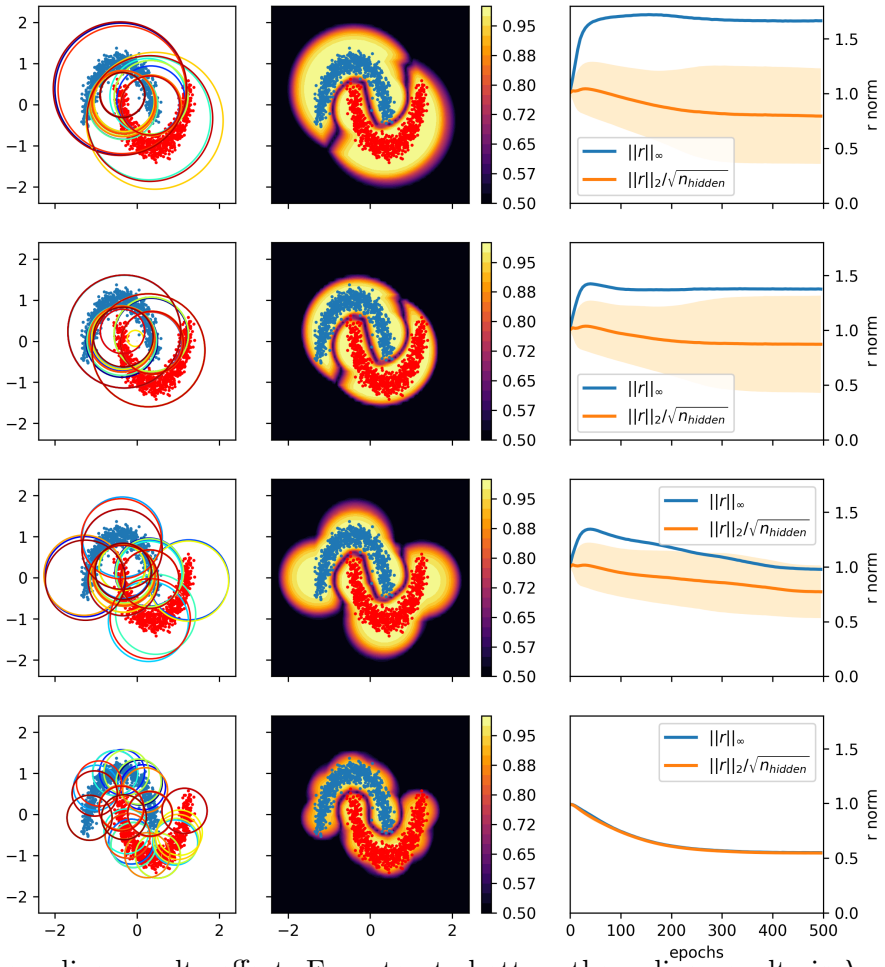

Figure 4.4: The radius penalty effect. From top to bottom the radius penalty is $\lambda = 0.0$, $\lambda = 0.02$, $\lambda = 0.04$ and $\lambda = 0.64$ respectively. From left to right are shown the support circles in the input space, the prediction confidence, and the evolution of the radius, in which the maximum and average radius $\pm$ std among 64 compact support neurons are plotted.

against the false positive rate FPR,

$$TPR = \frac{tp}{tp + fn} \,, \tag{4.2}$$

$$FPR = \frac{fp}{fp + tn} \,, \tag{4.3}$$

where tp indicates the number of true positives predicted, fn is the number of false negatives, fp represents the number of false positives, and tn stands for the number of true negatives predicted. AUROC is the area under the ROC curve. AUROC can be interpreted as the probability of placing

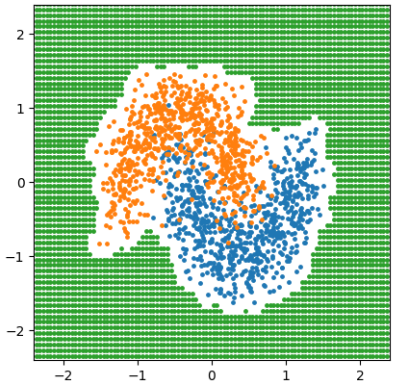

Figure 4.5: Samples in the moons dataset. The blue and orange dots are in-distribution samples, and the green dots are OOD samples.

a random positive sample on the right-hand side of a negative sample, e.g., a perfect model will output a 100% AUROC and a random predictor will output a 50% AUROC [Fawcett, 2006].

The test accuracy for in-distribution samples, the ratio of non-zero outputs for OOD samples, and the AUROC are plotted as functions of the training epochs in Fig. 4.6. There is a dip as the model transits from MLP to CSNN at the beginning stage of training but the model quickly adapts to the compact support and an immediate increase of test accuracy is observed. After about 30 epochs' training, the ratio of non-zero outputs decreases and the AUROC increases, while the test accuracy is largely the same. Minimizing the cross entropy loss will improve the test accuracy while minimizing the radius penalty will force the model to decrease the support. The model eventually converges to an ideal scenario in which it fits the training samples nicely and maintains a tight support at the same time. We can conclude that the network with compact support still has the flexibility and representation capability for this classification task.

The distribution of confidence scores is given in Fig. 4.7. The OOD samples are plotted one out of ten for better appearance. For the majority of OOD samples (95.5%), all compact support neurons output zero, i.e., the OOD samples are located outside the neuron supports, which leads to a uniform prediction and lowest possible confidence. For those that have non-zero output, the confidences are moderate. As can be seen in Fig. 4.7, the confidence score successfully separates the in-distribution samples from the OOD samples and the majority of OOD samples have smaller

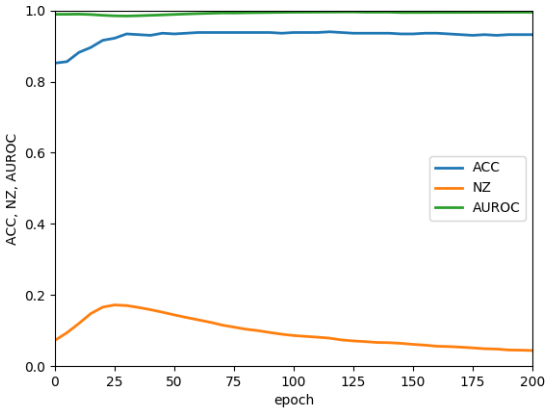

Figure 4.6: Convergence of test accuracy, non-zero output ratio, and AUROC.

confidences compared to the in-distribution samples. The ROC curve is given in Fig. 4.8. Due to the fact that the majority of OOD samples have smaller confidences compared to the in-distribution samples, the ROC curve hugs the top left corner, which leads to a very high AUROC score of 0.997.

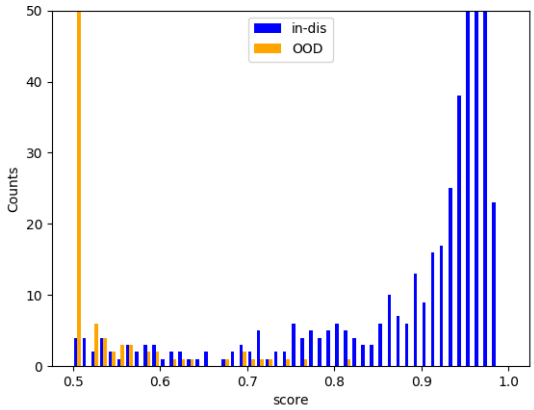

Figure 4.7: Histograms of confidences for in-distribution and OOD samples.

4.2 NGSIM Dataset

We will use the NGSIM dataset [Kovvali et al., 2007] for real data experiments. The extraction and annotation of human drivers' lane change behaviors are given in Chapter II. The detailed statistics of extracted lane changes are given in Table 3.1.

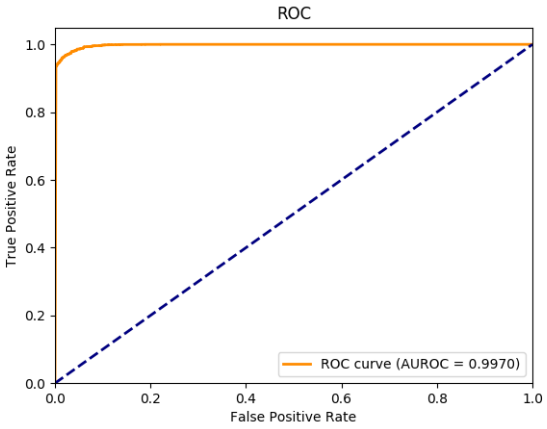

Figure 4.8: ROC curve of the moons dataset.

MLP networks are trained first on the NGSIM dataset and the test accuracy will be used as the baseline for in-distribution prediction performance. CSNN models will then be trained and the test accuracy will be compared with the MLP results. The in-distribution prediction accuracy and OOD detection performance will also be compared with results obtained from recently proposed OOD detection algorithms.

4.2.1 Feature Selection

As can be seen in Fig. 3.1, in carrying out a lane change, only the immediate leading vehicle in the old lane (v_2), the leading vehicle in the target lane (v_1) and the lag vehicle in the target lane (v_0) are relevant to ego. An instance at time t is represented with the following features

$$\boldsymbol{x} = [v_{ego},\ \Delta v_0,\ \Delta x_0,\ \Delta y_0,\ \Delta v_1,\ \Delta x_1,\ \Delta y_1,\ \Delta v_2,\ \Delta x_2,\ \Delta y_2]\ , \tag{4.4}$$

in which $\Delta v_i = v_{ego} - v_i$ are the relative velocities, $\Delta x_i = x_{ego} - x_i$ are the relative longitudinal positions and $\Delta y_i = y_{ego} - y_i$ are the relative lateral positions for $i \in \{0,\ 1,\ 2\}$. The features are extracted every $0.1s$ from $t_0 - 0.5$ to t_0 and averaged to get the final features. In such a way, the sensor noise is alleviated.

To facilitate the downstream OOD sample detection task, a backward feature selection method, i.e., dropping one feature at a time in a greedy fashion, is carried out using MLP models with two hidden layers with 64 neurons each. As can be seen in Fig. 4.9 and Table 4.1, using just 4 features

out of the 10 original features from (4.4),

$$\boldsymbol{x} = [\Delta v_0,\ \Delta x_0,\ \Delta v_1,\ \Delta x_1]\ , \tag{4.5}$$

the average test accuracy over 10 independent runs drops from 0.876 to 0.871. It is reasonable to assume that only these 4 features are strongly related to this prediction task, hence hereafter, this 4D feature space will be used.

Table 4.1: Backward Feature Selection

Feature dimension	Features dropped	Test accuracy
10	-	0.876
9	dy_2	0.88
8	dy_1	0.881
7	dy_0	0.88
6	v_{ego}	0.877
5	dx_2	0.875
4	dv_2	0.871
3	dv_1	0.862
2	dv_0	0.836

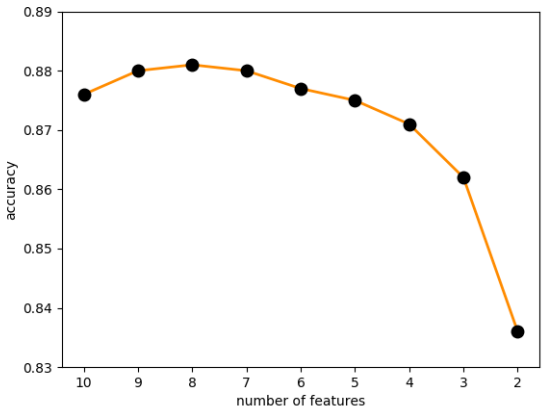

Figure 4.9: Best test set accuracy vs. number of selected features, averaged over 10 independent runs.

4.2.2 OOD Sample Generation

In previous studies, OOD samples are either are generated with a modified GAN (Generative Adversarial Network) approach of they are image datasets unrelated to the prediction task, e.g., when training on the MNIST dataset, which contains images of handwritten single digits, the OOD samples would be from the CIFAR10 dataset for example, which contains images of 10 object classes, such as dog and cat. In this task, we do not have an existing dataset that can serve as the OOD samples. Samples generated from modified GAN [Lee et al., 2018b] are found to be only around part of the low-density boundaries and do not cover the whole domain [Vernekar et al., 2019]. Since the dimension of feature space has been successfully decreased to $d = 4$, OOD samples can be generated through uniform sampling.

First, all the in-distribution samples, i.e., samples from both datasets (I-80 and US-101), are combined and normalized to have 0 mean and 1 std in each dimension.

Then for each in-distribution sample $\boldsymbol{x}_i$, the minimum distance d_i to other in-distribution samples is computed. Then we find a distance threshold τ as the 99 percentile of the d_i values, thus 99% of the d_i will be less than τ. The distribution of minimum distance to other in-distribution samples is shown in Fig. 4.10. As visualized by Fig. 4.10, the average minimum distance to other in-distribution samples in dataset US-101 is larger than in dataset I-80, i.e., samples in dataset I-80 are more concentrated. One potential reason is that there is much heavier traffic in dataset I-80, hence a majority of lane changes happen in a tight window.

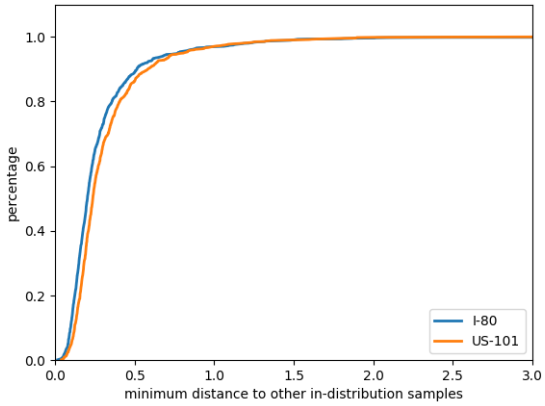

Figure 4.10: Minimum distance to other in-distribution samples.

Then we generate 160,000 samples through uniform sampling in the hyper-rectangle

$$\begin{aligned}\mathcal{R} =&[1.5\min_i \Delta v_0^i,\ 1.5\max_i \Delta v_0^i] \times [-r_0,\ 0.5r_0] \\ &\times [1.5\min_i \Delta v_1^i,\ 1.5\max_i \Delta v_1^i] \times [-0.5r_0,\ r_0]\ ,\end{aligned} \tag{4.6}$$

where $r_0 = 100$m is the detection range.

The generated samples are then transformed using the mean and std of the in-distribution samples and the minimum distance to any in-distribution samples is calculated. The generated samples with a distance $d_i < \tau$ are discarded. This way, 147,496 generated samples are kept as OOD samples. In comparison, there are 2,848 in-distribution samples.

4.2.3 Architectures

For the baseline, MLP models with 2 hidden layers of 64 hidden neurons are trained. To have a fair comparison, the CSNN models have the same number of layers and neurons except that the neurons in the last hidden layer are compact support neurons. There is also a batch normalization layer [Ioffe and Szegedy, 2015] installed right before the CSNN layer. There is no learnable parameters in the batch normalization layer, i.e.,

$$y = \frac{x - E[x]}{\sqrt{Var[x] + \epsilon}} * \gamma + \beta\ , \tag{4.7}$$

in which $\gamma = 1/\sqrt{d}$, d is the number of neurons in the layer right before the CSNN layer, $\beta = 0$, and $\epsilon = 10^{-5}$ is added to the denominator for numerical stability. The normalization will set the variables in inputs $\boldsymbol{x}$ of compact support neurons to be on the same scale and $||\boldsymbol{x}||$ to be approximately 1 assuming variables of $\boldsymbol{x}$ are d independent Gaussian random variables with 0 mean and 1 standard deviation. There is no bias term for the CSNN layer, such that the logits will be 0 when all the compact support neurons don't fire, i.e., when the testing samples are located outside the compact supports.

4.2.4 Training

Samples from I-80 dataset and US-101 dataset are combined as the in-distribution samples, in which 75% samples are used for training and the remaining samples for test. The Adam optimization algorithm [Kingma and Ba, 2015], a first-order algorithm with adaptive estimates of first and second

order moments, is employed.

$$
\begin{aligned}
g &\leftarrow \frac{d\ell}{d_\theta}\,, && (4.8)\\
m &\leftarrow \beta_1 m + (1-\beta_1) g\,, && (4.9)\\
v &\leftarrow \beta_2 v + (1-\beta_2) g^2\,, && (4.10)\\
\theta &\leftarrow \theta - \alpha \frac{m/(1-\beta_1^t)}{\sqrt{v/(1-\beta_2^t)}+\epsilon}\,, && (4.11)
\end{aligned}
$$

where m denotes the first momentum of gradient with exponential decay rate $\beta_1 = 0.9$ and initial value $m_{init} = 0$, v indicates the second momentum of gradient with exponential decay rate $\beta_2 = 0.999$ and initial value $v_{init} = 0$. The term $\epsilon = 10^{-8}$ is added to the denominator to improve numerical stability, the learning rate is $\alpha = 0.0001$, and t is the time step. Due to the inclusion of moments, Adam is well suited for optimization with sparse gradients and can escape local optimum better than stochastic gradient descent.

To have a consistent result and reliable comparison, 10 independent runs are carried out for each algorithm with different random initializations and shuffles of the training samples. The shape parameter α is increased linearly from 0 to 1 as the epoch number increases from 1 to 1000, i.e., we begin with normal-type ReLU neurons and gradually shrink the support. The parameter R is initialized to 1 and is learnable. The radius penalty coefficient is set by a grid search over the range $\lambda \in [0,\ 2]$ and the best test accuracy is obtained at $\lambda = 0.1$.

4.2.5 Methods Compared

The most similar OOD detection approach to ours is the Deterministic Uncertainty Quantification (DUQ) [Van Amersfoort et al., 2020] algorithm, which is also based on the RBF network and does not need OOD samples in training the classifier. To have a fair comparison, a MLP with 1 hidden layer of 64 units is used to extract the feature vector and the centroid is of size 64. The length scale σ and gradient penalty coefficient λ are set by a grid search over the space $\sigma \in (0, 1.0]$ and $\lambda \in [0, 1.0]$.

Different from the CSNN algorithm, DUQ is based on the Gaussian RBF, i.e.,

$$
\begin{aligned}
r_c &= ||\boldsymbol{x} - \boldsymbol{x}_c||\,, && (4.12)\\
K_c &= e^{-(\epsilon r_c)^2}\,, && (4.13)
\end{aligned}
$$

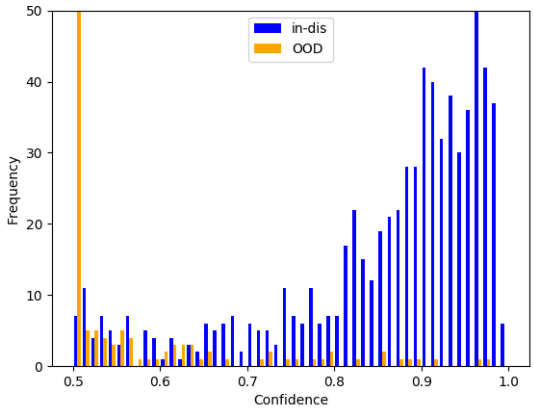

Figure 4.11: Histograms of confidences on the NGSIM dataset and the generated OOD samples.

where $\boldsymbol{x}_c$ is the class centroid, ϵ is the shape parameter, and K_c is the prediction confidence. DUQ has a strong assumption that each class has its own class centroid in the hidden space, while CSNN only assumes each neuron in the CSNN layer has its own support. There are also two hyperparamters, the gradient penalty coefficient λ and the length scale σ, while in CSNN, there is only one hyperparameter, the radius penalty λ, in our current setting of training.

Another method we compared with is the deep ensemble, which enjoys great success in high-dimensional applications [Ovadia et al., 2019, Gustafsson et al., 2020]. Models with the same architecture but different random initializations have been experimentally demonstrated to tend to disagree on OOD samples, hence leading to a higher entropy in prediction. In this study, ensembles are constructed from 10 independent nets trained with different initializations and different shuffles of the training samples. To have a fair comparison, each net is a MLP network with two layers of 64 hidden units. The biggest disadvantage of the deep ensemble approach is that the computation is heavy as the complexity is of order $O(n)$, where n is the number of networks. This shortcoming is alleviated with the help of modern high-efficiency parallel computing, through which all the networks can be trained and inferred at the same time.

4.2.6 Results

For the standard MLP baseline, the best test accuracy obtained by averaging 10 independent runs is 0.871 ± 0.002, in comparison, the positive rate of the testing set is 0.782.

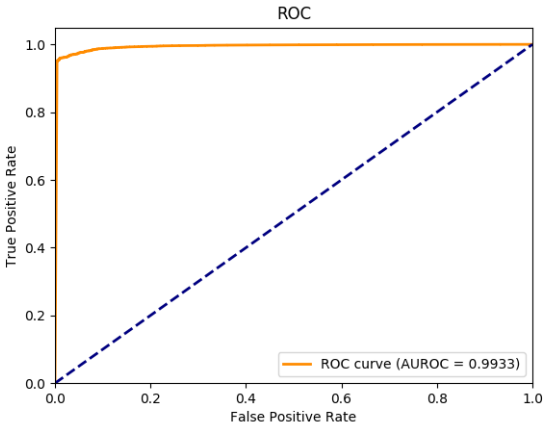

Figure 4.12: OOD detection ROC curve for one run of training the CSNN on the NGSIM dataset.

For the CSNN algorithm, the histograms of confidences of the in-distribution and OOD samples for one run are plotted in Fig. 4.11. The majority of OOD samples are locate outside the compact supports, hence the neurons in the penultimate layer don't fire, which in turn leads to the lowest confidence in this binary classification setting. The corresponding ROC curve for this run, which is depicted in Fig. 4.12, hugs the top left corner, and therefore has a very high AUROC of 0.993 is obtained. The average test accuracy and AUROC over 10 independent runs are plotted in Fig. 4.13 as functions of α.

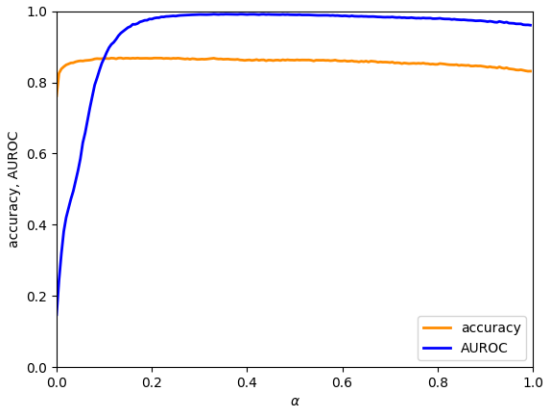

Figure 4.13: Test accuracy and AUROC vs α for the CSNN.

As we can see in Fig. 4.13, the test accuracy increases as α increases. The AUROC in detecting OOD samples also increases as α increases and the support becomes more compact. When the α fur-

Table 4.2: Test accuracy and AUROC in OOD detection, averaged of 10 independent runs.

	Test accuracy	AUROC
MLP	0.871 (.002)	0.156 (.012)
CSNN	0.868 (.002)	0.991 (.001)
DUQ	0.868 (.002)	0.971 (.005)
Deep ensemble	0.873 (.001)	0.189 (.005)

ther increases, both the test accuracy and AUROC decrease gently. The best AUROC 0.991 ± 0.001 is obtained at $\alpha = 0.33$, where test accuracy is 0.868 ± 0.002. Compared with the baseline result 0.871 ± 0.002 obtained with the normal neuron-based network, there is only a 0.3% decrease, i.e., CSNN's in-distribution prediction performance is comparable to a typical neuron-based network.

The best average test accuracy over 10 independent runs for DUQ is 0.868 ± 0.002, obtained at $\sigma = 0.4$ and $\lambda = 0.3$, where the AUROC in detecting OOD samples is 0.971 ± 0.005. As shown in Table 4.2, CSNN achieves the same in-distribution prediction accuracy compared with DUQ but outperforms DUQ in detecting OOD samples in this task. CSNN is also more robust compared to DUQ as the standard deviation in OOD detection is much smaller than DUQ's. One explanation is that CSNN has more flexibility and makes fewer assumptions as each neuron has its own support, while in DUQ, each class is assumed to have its own centroid.

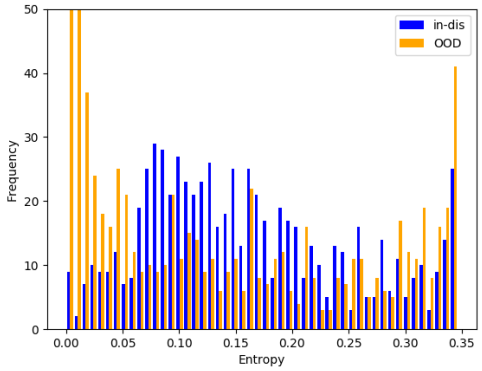

Figure 4.14: Entropy distribution of in-distribution and OOD samples.

The best average test accuracy of the deep ensemble is 0.873 ± 0.001, where the AUROC is 0.189 ± 0.005. The distribution of entropy scores for one run is given in Fig. 4.14 and corresponding

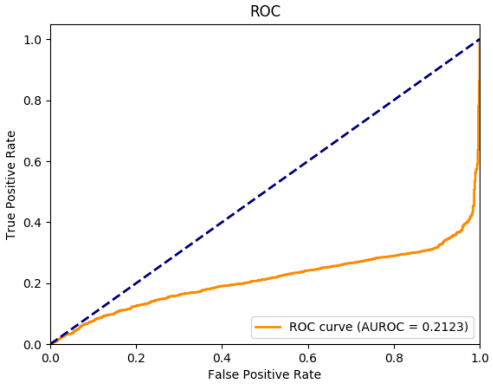

Figure 4.15: ROC curve obtained with deep ensemble.

ROC curve is visualized in Fig. 4.15. The deep ensemble completely fails in this low-dimensional OOD prediction task. The majority of OOD samples actually have smaller entropy compared with in-distribution samples, i.e., the individual networks actually tend to agree with each other on the OOD samples, which contradicts the assumption. The ROC curve is below the diagonal line and the AUROC is well below 0.5, while a random guessing classifier will achieve an AUROC of 0.5. Similarly, the DUQ paper [Van Amersfoort et al., 2020] also reports that deep ensembles do not work in low-dimensional applications for the OOD detection task.

To reveal the reasons, the entropy of the average prediction of the deep ensemble in another low-dimensional ODD detection task, the moons dataset, is plotted in Fig. 4.16. In contrast to the high-dimensional applications like image classification, in the low-dimensional scenario, the nets tend to only disagree with each other on the class overlap and near the decision boundary, i.e., instead of the distributional uncertainty, deep ensembles actually capture the data uncertainty in the low-dimensional application.

4.2.7 Generalization of CSNN

The generalization capability of the CSNN algorithm is investigated by training on dataset I-80 and testing on dataset US-101 and vice versa. We first train MLP networks and set the test accuracy as the baseline. CSNN algorithms are then trained and the test accuracy and AUROC are given in Table 4.3. We also include the results from DUQ algorithm and deep ensemble.

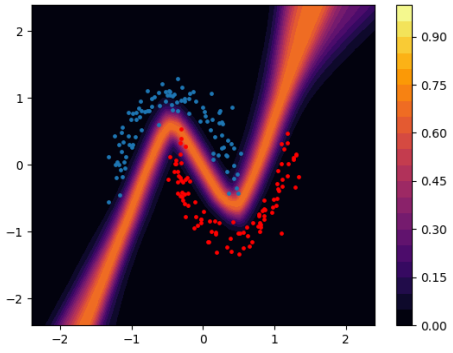

Figure 4.16: Entropy of average prediction.

Table 4.3: Generalization of Network with Compact Support

	Train I-80, test US-101		Train US-101, test I-80	
	Test accuracy	AUROC	Test accuracy	AUROC
MLP	0.925 (.003)	0.158 (.020)	0.8 (.004)	0.102 (.007)
CSNN	0.923 (.003)	0.993 (.002)	0.797 (.003)	0.974 (.003)
DUQ	0.923 (.001)	0.982 (.005)	0.809 (.003)	0.972 (.007)
Deep ensemble	0.924 (.001)	0.177 (.009)	0.803 (.002)	0.145 (.006)

When CSNN is trained on dataset I-80 and tested on dataset US-101, the test accuracy decreases negligibly compared with normal neuron-based networks. The AUROC in detecting OOD samples is still very high. When CSNN is trained on dataset US-101 and tested on dataset I-80, again the test accuracy is maintained compared with the normal neuron-based networks. There is a noticeable downgrade in OOD detection, however, i.e., some samples from dataset I-80 are recognized as OOD samples. We can conclude that there are more representative behaviors in dataset I-80 than in dataset US-101 and the behaviors demonstrated in dataset US-101 are a subset of behaviors demonstrated in dataset I-80. Another interpretation is that CSNN successfully detects the data drift in the dataset and separates the unseen novel samples.

4.3 Conclusion

In this chapter, we first demonstrated the effectiveness of the CSNN model in uncertainty quantification and OOD sample detection on a synthetic dataset. The CSNN model only has

support encompassing the training dataset, hence the model will not overgeneralize to the unseen domain. We then tested the influence of the radius penalty on tightening the support of the neurons, and found that with sufficient radius penalty, the support will tightly mimic the training data.

CSNN models are then trained on the NGSIM dataset. Random samples are generated with uniform sampling, spawning a large hyper-rectangle and the samples away from in-distribution samples are kept as OOD samples. Compared with the baseline results obtained with normal neuron-based networks, the CSNN models achieved comparable in-distribution prediction accuracy. Moreover, the CSNN models obtained very high AUROC in detecting OOD samples. The performance of the CSNN algorithm is compared with state-of-the-art algorithms, i.e., deep ensemble and DUQ. Similar to CSNN, DUQ has roots in the RBF network. CSNN obtained the same in-distribution prediction accuracy as DUQ, but surpassed DUQ in OOD sample detection. The CSNN models are also more robust compared with DUQ in OOD sample detection as the std of AUROC is much smaller for the CSNN models. Even though deep ensembles enjoy great success in high-dimensional applications, i.e., image classification, they completely failed in this low-dimensional task. The argument that different networks tends to disagree with each other on OOD samples is false in low-dimensional applications. The entropy of the average prediction is high on the decision boundary, i.e., deep ensemble captures data uncertainty in low-dimensional applications. By training on dataset I-80 and testing on dataset US-101, and vice versa, differences in driving behaviors in the two datasets are demonstrated. There is heavier traffic in dataset I-80 and the driving behaviors are more representative compared with dataset US101.

With compact support, not only can the model predict whether it will be an appropriate lane change, but the model can also detect the novelty of the samples fed in. If the samples are different from the training samples, the model will output a low-confidence prediction. The model can then either alert human operators to take over or suggest a conservative action.

CHAPTER 5

CONCLUSION

In this dissertation, we proposed a robust machine learning method for predicting human drivers' lane change decisions using Compact Support Neural Networks. We defined robust machine learning as a machine learning algorithm that can detect the novelty of samples and separate OOD samples from in-distribution samples. By introducing a shape parameter, the CSNN elegantly interpolates between the ReLU-type NN and a traditional RBF network. The neurons with compact support will fire if and only the input is within the sphere of support.

In the foreseeable future, autonomous vehicles will have to drive alongside human drivers. Without vehicle-to-vehicle communication, autonomous vehicles have to infer other road users' intentions and act like typical human drivers so that other road users can anticipate autonomous vehicles' actions. We proposed to learn human drivers' mental models in lane changes from the naturalistic driving dataset NGSIM. We extracted the lane changes from the dataset. Based on the deceleration of lag vehicles in the target lane during the entire lane change process and the window preferences, we labeled the extracted lane changes as cooperative and adversarial lane changes.

We experimentally demonstrated the effectiveness of CSNN model in detecting OOD samples on the synthetic Moons dataset. We further tightened the support by introducing the radius penalty. We then experimented on the CSNN dataset. We generated random samples through uniform sampling spawning a large hyper-rectangle and kept the ones far from training samples as OOD samples. We then trained CSNN models to predict the lane change behaviors and experimentally demonstrated that the trained models have comparable in-distribution prediction accuracy compared with normal neuron-based networks. The model achieved an AUROC of 0.991 in separating the OOD samples from in-distribution samples. We also compared the in-distribution prediction accuracy and OOD detection performance with recently developed OOD sample detection methods, i.e., deep ensemble and DUQ. The CSNN obtained the same in-distribution prediction accuracy but outperformed DUQ in OOD sample detection. The deep ensemble failed completely in OOD sample detection. We computed the entropy of the average prediction and found that the entropy

is large on the decision boundaries. In low-dimensional applications, the individual networks actually tend to disagree on the class overlap, i.e., the ensemble captures data uncertainty instead of distributional uncertainty.

The trained model can be integrated into the P&C module of an autonomous vehicle and the vehicle will mimic human beings' driving behaviors in lane changes, i.e., only carries out a lane change when most human drivers consider it is appropriate. The model can also separate the unseen novel samples from the training dataset, which is of paramount value in deploying machine learning models to safety-critical applications like autonomous driving. The perception system of self-driving vehicles is backed by machine learning models, especially deep learning models, which are badly calibrated. Overconfident prediction and overgeneralization will lead to unpredictable and dangerous behaviors. We hope this work will inspire researches in uncertainty quantification and OOD sample detection in the self-driving industry.

In the future, we will try to separate the distributional uncertainty from the data uncertainty. Currently the model will output low confidence when the sample is far from the training dataset or when there is class overlap. Uncertainty from class overlap is arguably less risky compared with distributional uncertainty as human drivers will do both of them, while for distributional uncertainty, we simply do not know what might happen.

BIBLIOGRAPHY

A. Alizadeh, M. Moghadam, Y. Bicer, N. K. Ure, U. Yavas, and C. Kurtulus. Automated lane change decision making using deep reinforcement learning in dynamic and uncertain highway environment. In *ITSC*, pages 1399–1404, 2019.

D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

E. Balal, R. L. Cheu, and T. Sarkodie-Gyan. A binary decision model for discretionary lane changing move based on fuzzy inference system. *Transp. Res. C: Emerging Technologies*, 67: 47–61, 2016.

A. Barbu and H. Mou. The compact support neural network. *http://arxiv.org/abs/2104.00269*, 2021.

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015.

S. Brechtel, T. Gindele, and R. Dillmann. Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps. In *17th international IEEE conference on intelligent transportation systems (ITSC)*, pages 392–399. IEEE, 2014.

D. S. Broomhead and D. Lowe. Radial basis functions, multi-variable functional interpolation and adaptive networks. Technical report, Royal Signals and Radar Establishment Malvern (United Kingdom), 1988.

Y. Chen, C. Dong, P. Palanisamy, P. Mudalige, K. Muelling, and J. M. Dolan. Attention-based hierarchical deep reinforcement learning for lane change behaviors in autonomous driving. In *IROS*, 2019.

C. Cortes, G. DeSalvo, and M. Mohri. Learning with rejection. In *International Conference on Algorithmic Learning Theory*, pages 67–82. Springer, 2016.

S. Depeweg, J.-M. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pages 1184–1193. PMLR, 2018.

K. Driggs-Campbell, V. Govindarajan, and R. Bajcsy. Integrating intuitive driver models in autonomous planning for interactive maneuvers. *IEEE Transactions on Intelligent Transportation Systems*, 18(12):3461–3472, 2017.

T. Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

Y. Geifman and R. El-Yaniv. Selective classification for deep neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4885–4894, 2017.

Y. Geifman and R. El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. In *International Conference on Machine Learning*, pages 2151–2159. PMLR, 2019.

P. G. Gipps. A model for the structure of lane-changing decisions. *Transportation Research Part B: Methodological*, 20(5):403–414, 1986.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014a.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

A. Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356. Citeseer, 2011.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *ICML*, pages 1321–1330, 2017.

F. K. Gustafsson, M. Danelljan, and T. B. Schon. Evaluating scalable bayesian deep learning methods for robust computer vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 318–319, 2020.

D. Hafner, D. Tran, T. Lillicrap, A. Irpan, and J. Davidson. Noise contrastive priors for functional uncertainty. In *Uncertainty in Artificial Intelligence*, pages 905–914. PMLR, 2020.

M. Hein, M. Andriushchenko, and J. Bitterwolf. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *CVPR*, pages 41–50, 2019.

D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. 2017.

D. Hendrycks, M. Mazeika, and T. Dietterich. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2018.

P. Hidas. Modelling vehicle interactions in microscopic simulation of merging and weaving. *Transportation Research Part C: Emerging Technologies*, 13(1):37–62, 2005.

C.-J. Hoel, K. Wolff, and L. Laine. Automated speed and lane change decision making using deep reinforcement learning. In *ITSC*, pages 2148–2155, 2018.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2): 251–257, 1991.

Y.-C. Hsu, Y. Shen, H. Jin, and Z. Kira. Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10951–10960, 2020.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

S.-G. Jeong, J. Kim, S. Kim, and J. Min. End-to-end learning of image based lane-change decision. In *IV*, pages 1602–1607, 2017.

H. Jiang, B. Kim, M. Y. Guan, and M. R. Gupta. To trust or not to trust a classifier. In *NeurIPS*, pages 5546–5557, 2018.

A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NIPS*, 2017.

A. Kesting, M. Treiber, and D. Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

H. Kita. A merging–giveway interaction model of cars in a merging section: a game theoretic analysis. *Transportation Research Part A: Policy and Practice*, 33(3-4):305–312, 1999.

V. G. Kovvali, V. Alexiadis, and L. Zhang PE. Video-based vehicle trajectory data collection. In *Transp. Res. Board Annual Meeting*, 2007.

R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein. The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2118–2125. IEEE, 2018.

V. Kuleshov and P. S. Liang. Calibrated structured prediction. *Advances in Neural Information Processing Systems*, 28:3474–3482, 2015.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018a.

K. Lee, K. Lee, H. Lee, and J. Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018b.

S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.

J. Liu, Z. Lin, S. Padhy, D. Tran, T. Bedrax Weiss, and B. Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *NeurIPS*, 33, 2020a.

W. Liu, X. Wang, J. Owens, and Y. Li. Energy-based out-of-distribution detection. *NeurIPS*, 33, 2020b.

A. Malinin and M. Gales. Predictive uncertainty estimation via prior networks. In *NeurIPS*, 2018.

A. Meinke and M. Hein. Towards neural networks that provably know when they don't know. In *International Conference on Learning Representations*, 2019.

B. Mirchevska, C. Pek, M. Werling, M. Althoff, and J. Boedecker. High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning. In *ITSC*, 2018.

S. Mohseni, M. Pitale, J. Yadawa, and Z. Wang. Self-supervised learning for generalizable out-of-distribution detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5216–5223, 2020.

V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, pages 427–436, 2015.

NTSB. Collision between vehicle controlled by developmental automated driving system and pedestrian. Technical report, 2019.

Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *arXiv preprint arXiv:1906.02530*, 2019.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *JMLR*, 12: 2825–2830, 2011.

M. Rahman, M. Chowdhury, Y. Xie, and Y. He. Review of microscopic lane-changing models and future research opportunities. *IEEE transactions on intelligent transportation systems*, 14(4): 1942–1956, 2013.

J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *NeurIPS*, pages 14680–14691, 2019.

O. Scheel, L. Schwarz, N. Navab, and F. Tombari. Situation assessment for planning lane changes: Combining recurrent models and prediction. In *ICRA*, pages 2082–2088, 2018.

O. Scheel, N. S. Nagaraja, L. Schwarz, N. Navab, and F. Tombari. Attention-based lane change prediction. In *ICRA*, pages 8655–8661, 2019.

P. Schulam and S. Saria. A framework for individualizing predictions of disease trajectories by exploiting multi-resolution structure. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, pages 748–756, 2015.

W. Schwarting, J. Alonso-Mora, and D. Rus. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 2018.

T. Shi, P. Wang, X. Cheng, C.-Y. Chan, and D. Huang. Driving decision and control for automated lane change behavior based on deep reinforcement learning. In *ITSC*, pages 2895–2900, 2019.

J. Snoek, Y. Ovadia, E. Fertig, B. Lakshminarayanan, S. Nowozin, D. Sculley, J. Dillon, J. Ren, and Z. Nado. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pages 13969–13980, 2019.

S. Sun, G. Zhang, J. Shi, and R. Grosse. Functional variational bayesian neural networks. In *International Conference on Learning Representations*, 2018.

D. M. Tax and R. P. Duin. Growing a multi-class classifier with a reject option. *Pattern Recognition Letters*, 29(10):1565–1570, 2008.

S. Ulbrich and M. Maurer. Towards tactical lane change behavior planning for automated vehicles. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 989–995. IEEE, 2015.

J. Van Amersfoort, L. Smith, Y. W. Teh, and Y. Gal. Uncertainty estimation using a single deep deterministic neural network. In *ICML*, pages 9690–9700, 2020.

S. Vernekar, A. Gaurav, T. Denouden, B. Phan, V. Abdelzad, R. Salay, and K. Czarnecki. Analysis of confident-classifiers for out-of-distribution detection. *arXiv preprint arXiv:1904.12220*, 2019.

M. Wang, S. P. Hoogendoorn, W. Daamen, B. van Arem, and R. Happee. Game theoretic approach for predictive lane-changing and car-following control. *Transportation Research Part C: Emerging Technologies*, 58:73–92, 2015.

B. Wolshon and A. Pande. *Traffic engineering handbook*. John Wiley & Sons, 2016.

D.-F. Xie, Z.-Z. Fang, B. Jia, and Z. He. A data-driven lane-changing model based on deep learning. *Transp. res. C: emerging technologies*, 106:41–60, 2019.

Z. Yan, J. Wang, and Y. Zhang. A game-theoretical approach to driving decision making in highway scenarios. In *IV*, pages 1221–1226, 2018.

H. Yu, H. E. Tseng, and R. Langari. A human-like game theory-based controller for automatic lane changing. *Transportation Research Part C: Emerging Technologies*, 88:140–158, 2018.

X. Zhang, J. Sun, X. Qi, and J. Sun. Simultaneous modeling of car-following and lane-changing behaviors using deep learning. *Transpp. res. C: emerging technologies*, 104:287–304, 2019.

BIOGRAPHICAL SKETCH

Hua Huang obtained a B.S. degree from Xi'an Jiaotong University and an M.S. degree from Shanghai Jiao Tong University, both of them in aerospace engineering. Hua went on to pursue a Ph.D. degree in Applied and Computational Mathematics at Florida State University.

Hua got exposed to the great success of machine learning in large-scale industrial applications in his internship at Lawrence Berkeley National Laboratory working as a computational science intern. He later joined Dr. Adrian Barbu's research group and worked on reinforcement learning. He also worked as a machine learning research intern at a self-driving startup, during which he prototyped the application of reinforcement learning to model lane changes. The idea of robust machine learning caught his attention after the internship project. He then worked on uncertainty quantification and out-of-distribution sample detection. During his time at Florida State University, he worked as a teaching assistant and taught college algebra and calculus.