Proceedings of the 2023 Winter Simulation Conference
C. G. Corlu, S. R. Hunter, H. Lam, B. S. Onggo, J. Shortle, and B. Biller, eds.

UNCERTAINTY-QUANTIFIED, ROBUST DEEP LEARNING FOR NETWORK INTRUSION DETECTION

Joshua A. Wong

Department of Mathematical Sciences
United States Military Academy
606 Thayer Road Road
West Point, NY 10996, USA

Alexander M. Berenbeim
David A. Bierbrauer
Nathaniel D. Bastian

Army Cyber Institute
United States Military Academy
2101 New South Post Road
West Point, NY 10996, USA

ABSTRACT

Cyber threats are moving beyond human comprehension and reaction capability in a rapidly evolving world. Deep learning models for network intrusion detection are becoming evermore crucial in processing network traffic to filter benign content from malicious activity. However, novel attacks such as zero-days are becoming more frequent, demonstrating the need for robust deep learning models to flag attacks while providing predictive certainty guarantees. Therefore, detecting out-of-distribution (OOD) inputs at inference time is crucial to address the rapidly changing environment while keeping up with evolving cyber threats. We develop multi-class deep learning models for network intrusion detection, comparing deterministic with Bayesian neural networks estimated using Hamiltonian Monte Carlo. We also propose new uncertainty quantification scoring measures for performance evaluation to evaluate certainty in predictions. During our experimentation, our best performing proposed Bayesian deep learning model detected 89.1% and 86.9% of the OOD packets at the 5% and 0.1% significance levels, respectively.

1 INTRODUCTION

Cybersecurity is increasingly critical as networks and data increase in size and complexity. With the increasing reliance on technology and the internet, personal and organizational data is being generated and stored digitally, making it vulnerable to cyber threats. Large networks that hold valuable data provide incentives for malicious actors to develop creative ways to compromise the security of private networks. In light of a growing number of sophisticated cyber attacks (Jalali et al. 2019; Franco et al. 2020) , it is critical for organizations and individuals to stay up-to-date on healthy cybersecurity practices and to implement robust cybersecurity measures to protect their data.

In the face of significant cybersecurity threats, organizing and processing related security data is critical to creating data-driven, intelligent solutions to cybersecurity problems (Sarker et al. 2020; Coulter et al. 2019). One method contributing to network security is using network intrusion detection systems (NIDS). NIDS identify unauthorized use, misuse, and abuse of computer systems by users inside and outside the network. Due to the increasing threat of malicious cyber actors, security professionals prefer adaptive security measures. As attackers develop new tactics, techniques, and procedures to compromise private networks and sensitive data, robust models become vital in identifying malicious network activity.

Cyber threats are moving beyond human comprehension and reaction capability in a rapidly evolving modern world. Machine learning models are crucial in processing network data to filter benign content from malicious activity (Coulter et al. 2019). However, as the cyber battlefield space rapidly changes, zero-day attacks are becoming more frequent, demonstrating the need for robust machine learning models. Therefore,

detecting out-of-distribution (OOD) inputs is crucial to address the rapidly changing environment. Current machine learning methods produce deterministic outputs and train well on existing datasets. However, these models fail to succeed in real-world applications when faced with novel attacks and zero-day exploits (Bilge and Dumitraş 2012). Detecting OOD attacks is essential to keeping up with evolving cyber threats. Applying uncertainty quantification, Bayesian deep learning models can learn to detect OOD inputs by incorporating uncertainty quantification to provide certainty levels for predictions. A model calculates these certainty levels for predictions using the probability distribution of the Bayesian deep learning model output, which leads to low certainty scores being associated with input data that is considered OOD.

This work compares the results from four multi-class deterministic deep learning models to four multi-class Bayesian (stochastic) deep learning models that utilize Hamiltonian Monte Carlo (HMC) for uncertainty quantification. We create the four models as a Fully-Connected Neural Network or a One-Dimensional Convolutional Neural Network, trained on either of two open source data data sets, and developed a novel uncertainty quantification score that describes each models certainty in a prediction. The main contributions of this research are: 1) the deep learning models, trained on raw payload bytes, that implement uncertainty quantification for model predictions for network intrusion detection; and 2) verification that independent of architecture choice, the certainty scores of True Positives and False Positives are drawn from different distributions, such that in the stochastic setting we can use the distribution of a packet's certainty scores to detect misclassification and OOD inputs. The Bayesian models provide confidence levels with predictions, allowing these models to detect novel OOD inputs. Uncertainty quantified, robust deep learning models such as these are essential for real-world applications due to the fast-paced evolution of cyber threats.

The paper is structured as follows. Section 2 briefly reviews related works on deep learning models for network intrusion detection and uncertainty quantification. Section 3 discusses the research methodology, specifically the data sets and models developed and evaluated during computational experimentation. Section 4 outlines our results, whereas Section 5 summarizes our conclusions, limitations, and future work.

2 LITERATURE REVIEW

2.1 Deep Learning for Network Intrusion Detection

Artificial intelligence and machine learning techniques for network intrusion detection are rapidly advancing. Deep learning accounts for most of the research surrounding machine learning techniques for network intrusion detection (Ahmad et al. 2021; Srivastava and Richhariya 2013; Kumar et al. 2021). Some deep learning methods for network intrusion detection include Variational Autoencoders, Deep Neural Networks, Convolutional Neural Networks, Recurrent Neural Networks, and Belief Neural Networks. While most models produce high-accuracy measurements, most are deterministic (Potluri and Diedrich 2016; Liu and Lang 2019), and only limited number evaluated their model's ability to detect OOD data (Bradley et al. 2023). There are multiple papers applying machine learning techniques to network intrusion detection. However, research into uncertainty quantification for NIDS is lacking.

2.2 Netflow Versus Packet Capture Data

Most current researchers use the same simulated Netflow datasets to train their machine learning models. A few benchmark datasets used for network intrusion detection are KDD Cup'99, Kyoto 2006+, NSL-KDD, and CSE-CICIDS-2018 (Ahmad et al. 2021; Shone et al. 2018; Kim et al. 2017; Liu and Lang 2019; Almseidin et al. 2017; Almiani et al. 2020). These datasets are created by simulating a network and attacks on the network in an artificial environment. The problem with Netflow data is that communication between two devices must end before the environment collects data, not allowing for in-time predictions. NetFlow data provides a high-level view of network traffic and can span various amounts of time per communication session, and contains signature-based data, such as internet protocol(IP) addresses, port numbers, and protocol types.

A less explored area of network data is in the individual packets that make up NetFlow reports, which would provide a raw, unprocessed picture of the network at a granular level. Researchers have investigated the effectiveness of packet-level data for network intrusion detection and found success with this granular approach (Zhang et al. 2019). A key benefit of analyzing packets is that due to the frequency of packets through the network, models have the potential to detect threats in real-time, compared to Netflow data which requires communication to end before a model can detect malicious activity. Papers discussed similar methods for creating and prepossessing packet-level data, converting each byte into an integer from 0 to 255 and filling in missing bytes with 0s to feed into a neural network (Farrukh et al. 2022; Zhang et al. 2019). Deep learning applications for network intrusion detection using packet-level data provide promising results with high accuracy (Zhang et al. 2019; Yu et al. 2021; Bierbrauer et al. 2023). However, previous models developed from training on packet-level data did not look into detecting OOD inputs.

2.3 Uncertainty Quantification Methods

Models must be robust enough to provide confidence levels with their predictions to detect OOD inputs. Stochastic models are critical to detecting potential zero-day exploits and separate in-distribution and OOD data. Various uncertainty quantification methods include Monte Carlo, Drop out, Markov Chain Monte Carlo, Variational Inference, Bayesian Active Learning, Bays by Backdrop, Variational autoencoders, Laplacian approximations, and Deep Ensembles (Abdar et al. 2021). Current studies see uncertainty quantification applied to a variety of different fields, including computer vision (Kohl et al. 2018), natural language processing (Chien and Ku 2016), and bioinformatics (Liu and Lang 2019; Abdar et al. 2021).

Bayesian neural networks provide context to model predictions through uncertainty quantification techniques and provide models with the tools to detect OOD inputs. While Bayesian neural networks demonstrate their usefulness in various fields, their application in cybersecurity, particularly for network intrusion, is lacking. Using Bayesian neural networks and uncertainty quantification in network intrusion detection would support a robust model to detect zero-day cyber-attacks and novel tactics from malicious actors. In particular, HMC is a method that simulates unknown probability distributions using Hamiltonian dynamics. The technique is a variation of the Metropolis-Hasting's algorithm with the added physical analogy (Betancourt 2018; Neal 2012). In our paper, we chose HMC for our stochastic models.

3 METHODOLOGY

This study compares the performance of two multi-class deep learning models, a Fully-Connected Neural Network (FcNN) and a One-Dimensional Convolutional Neural Network (1dCNN), implemented with and without HMC. Compared to previous work, our model aims to provide confidence levels for predictions and detect threats in real-time by training models on packet-level data. We trained each model using modified CICIDS-2017 and UNSW-NB15 datasets (Sharafaldin, Lashkari, and Ghorbani 2018; Moustafa and Slay 2015). Payload-Byte expanded the NetFlow data of two datasets into packet-level data, focusing on the packet's payload to limit the effect of signatures (Farrukh et al. 2022). In total, we train eight models, each with a different architecture, FcNN or 1dCNN, trained on two different datasets, and either with or without implementing HMC for uncertainty quantification.

3.1 Data

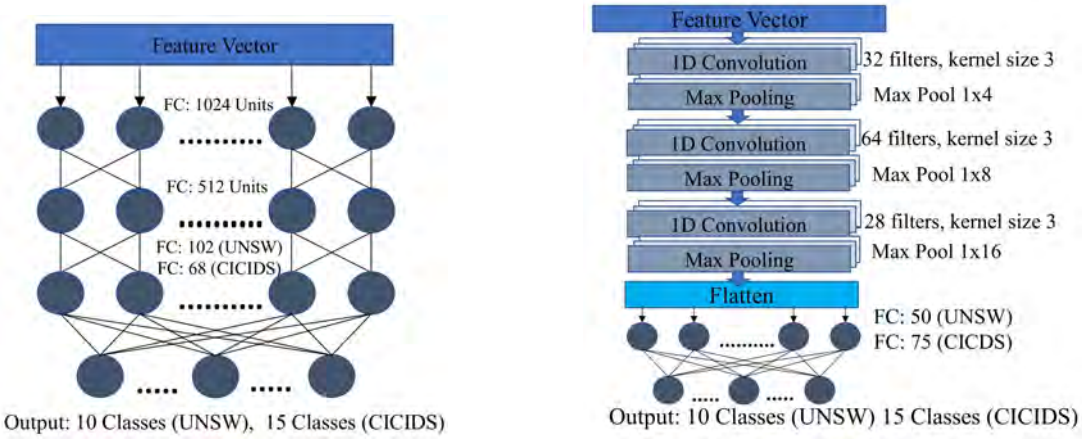
In this research, we use two popular datasets for NIDS deep learning: CICIDS-2017 (Sharafaldin et al. 2018) and UNSW-NB15 (Moustafa and Slay 2015). We chose these two datasets as in Bierbrauer et al. (2023) because they include raw packet capture data and labeled network flows. The Canadian Institute for Cybersecurity generated the CICIDS dataset containing five days of network traffic with various attacks. The UNSW dataset contains 100 GB of packets representing normal activity and synthetic attacks.

We use Payload-Byte, a standardized method using metadata to label raw packet captures for NIDS datasets (Farrukh et al. 2022), to label the packet capture data from the CICIDS and UNSW data sets.

Each data point in the modified CICIDS and UNSW datasets contained 1500 payload bytes with values of 0-255, a label, and various header information. If a packet was smaller than 1500 bytes, the rest of the bytes were filled in with zeros, and if a packet was longer, it was cut off at 1500 bytes. In this study, we dropped the header information and only focused on the packet payload and labels. We also normalized each byte in the data to a value between 0 and 1 to prepare the data for the models. The CICIDS data contained 15 classes, one benign class, and 14 attack types, and the UNSW data contained ten classes, one benign class, and nine attack types.

3.2 Deep Learning Architectures

We first implemented a FcNN to learn multi-class identification of the different attacks and benign traffic. The FcNN consisted of four dense layers in Figure 1a. The first and third layers used a ReLU activation function, while the second layer used a LeakyReLU activation function. The final layer used a SoftMax activation function for the output. The final two layers' sizes depended on the number of classes in the data set. We calculated the units in each model's second to last layer by dividing the size of the first layer by the number of classes in the data set. The final output layer of each model was sized depending on the number of classes in each data set. All the layers used random normal distribution for the kernel initializer. We compiled the FcNN using sparse categorical cross-entropy and an Adam optimizer.



(a) FcNN (b) 1dCNN

Figure 1: Deep learning model architecture.

We also implemented a 1dCNN depicted in Figure 1b. The intuition follows that the success of CNN models in natural language processing would translate to the language-like structure of the payload-byte data. The 1dCNN contained three 1D convolutional layers and three max pooling layers. The 1D convolutional layers used a Relu activation function. The filter size started at 32 for the first layer and was doubled for each subsequent layer. The pool size for the max pooling layer began at four and doubled for each subsequent layer. All convolutional and max pooling layers utilized the same padding and a stride size of one. After flattening the data, the model used two dense layers whose size depended on the number of classes in the data set. We determined the number of units in the first dense layer by multiplying the number of classes in the models' respective data sets by five. The first dense layer also used a ReLU activation function. The size of the final output layer was equal to the number of classes in the data set and used a softmax activation function. We compiled the 1dCNN model using sparse categorical cross-entropy and an Adam optimizer. It should be noted that we intentionally decided not to tune the hyperparameters of the different architectures to effectively compare the application of HMC in two different contexts represented

by the two different dataset models to demonstrate that our certainty scoring approach to OOD detection did not depend on our choice of architecture nor implementation in a deterministic or setting.

3.3 Hamiltonian Monte Carlo Simulation

Following the approach of Cobb et al. (2021), we also studied Bayesian neural networks with identical architectures to our deterministic FcNN and 1dCNN deep learning models estimated using HMC. Unlike deterministic neural networks, Bayesian neural networks require learning a distribution over the neural network weights, which we hypothesize can provide the additional benefit of OOD input detection for novel attacks. Bayesian neural networks are an important area of research, and many inference schemes may be deployed to learn the posterior distribution over the model parameters. In this paper, we sample directly from the posterior via Markov Chain Monte Carlo (MCMC) using HMC (Neal 1995) implemented in `hamiltorch` (Cobb and Jalaian 2020), which makes it feasible to run HMC on a single GPU.

HMC is a MCMC method that uses the gradients of the sampled density function to generate transitions spanning the posterior by employing Hamiltonian dynamics to traverse the parameter space of models. This occurs in two parts: first, an approximate Hamiltonian dynamics simulation based on numerical integration; then, a correction by performing a Metropolis acceptance step. The goal of the sampling is to draw from the Bayesian posterior $p(\boldsymbol{\omega} \mid \boldsymbol{Y}, \boldsymbol{X})$, for parameters $\boldsymbol{\omega}$ and data $\boldsymbol{X}, \boldsymbol{Y}$.

More concretely, to materialize samples HMC starts from the unnormalized log posterior, which is defined via the likelihood $p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\omega})$ and the prior $p(\boldsymbol{\omega})$ as follows:

$$p(\boldsymbol{\omega}|\mathbf{Y}, \mathbf{X}) \propto p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\omega})\, p(\boldsymbol{\omega}).$$

The likelihood is a function of the parameters, $\boldsymbol{\omega} \in \mathbb{R}^D$, which in our case will be the weights of the Bayesian neural network models. We then transform the Bayesian model into a Hamiltonian system by introducing a momentum variable $\mathbf{p} \in \mathbb{R}^D$, such that we now have a log joint distribution, $\log[p(\boldsymbol{\omega}, \mathbf{p})] = \log[p(\boldsymbol{\omega}|\mathbf{Y}, \mathbf{X})\, p(\mathbf{p})]$, that is proportional to the Hamiltonian, $H(\boldsymbol{\omega}, \mathbf{p})$. If we let $p(\mathbf{p}) = \mathcal{N}(\mathbf{p}|\mathbf{0}, \mathbf{M})$, where the covariance $\mathbf{M}$ denotes the mass matrix, our Hamiltonian can then be written as:

$$H(\boldsymbol{\omega}, \mathbf{p}) = \underbrace{-\log[p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\omega})\, p(\boldsymbol{\omega})]}_{\substack{\text{Potential Energy} \\ U(\boldsymbol{\omega})}} + \underbrace{{}^1\!/_2\ \mathbf{p}^\top \mathbf{M}^{-1} \mathbf{p}}_{\substack{\text{Quadratic Kinetic Energy} \\ K(\mathbf{p})}}.$$

We sample from this model by simulating the dynamics of the Hamiltonian system, using the leapfrog integration scheme– the details of this scheme are to be found in Neal et al. (2011), which provides a general overview, and Cobb and Jalaian (2020), which gives the specific symmetric scheme we used. Given the high-dimensionality of the packet data and our need to balance trade-offs between transition distance and mixing time, we found that HMC with No U-Turn Sampling (NUTS) was a more efficient method than Metropolis-Hastings and Gibbs sampling. HMC approaches areas of high density under the target distribution more rapidly than Metropolis-Hastings.

3.4 Sampling and Evaluation

We compare our two deterministic neural networks against their corresponding Bayesian neural networks. The weights of our deterministic models are optimized using stochastic gradient descent to determine a single model $\mathbf{f}(\cdot; \boldsymbol{\omega})$ with parameters $\boldsymbol{\omega}$, such that on a given input $\mathbf{x}^*$, our model outputs a prediction $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x}^*; \boldsymbol{\omega})$. For our Bayesian neural networks, we generate samples from the posterior distribution resulting in a set of samples $\{\boldsymbol{\omega}_s\}_{s \in S} \sim p(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{Y})$ such that the predictive distribution can be approximated via multiple networks draws for each test image $\hat{\mathbf{y}}_s = \mathbf{f}(\mathbf{x}^*; \boldsymbol{\omega})$. In particular, when evaluating our Bayesian neural networks, we compare the deterministic model against $\mathbf{f}_e$, an averaged ensemble model whose weights are determined after a burn-in period. Although a burn-in period is not absolutely necessary, using NUTS allowed us to optimize for the leap-frog step-size hyperparameter to target a constant Metropolis

rejection rate. This is because the different model architectures theoretically correspond to different density functions; our use of HMC across the different architectures was more assured due to the optimizing of this hyperparameter. In comparing our deterministic and Bayesian models, we directly compare two primary scores: accuracy and a novel score of certainty (Berenbeim et al. 2023). We further show how the ability to capture uncertainty results in more robust decision-making in the Bayesian context. While we used certainty scores to compare each model's ability to classify categories in aggregate, in contrast to the deterministic models, certainty scoring in the Bayesian models also allows for more descriptive certainty scoring on a packet level, which in turn allows for OOD detection and better model explainability.

3.5 Evaluation Metrics

During our experimentation, we rely on the following scores: multi-class classification accuracy, binary classification accuracy, false omission rate, misclassified positive rate, F1 score, and an additional score we call certainty. The multi-class classification accuracy gets after the model's ability to correctly classify packets as benign traffic or as one attack type. Binary classification accuracy only considers whether the packet was correctly classified as benign or malicious and is computed as the sum of True Positive (TP) malicious packets, and True Negative (TN) benign packets, divided by the total number of packets. One of our goals was to measure the model's performance in a way that unevenly punishes an attack misclassification compared to the misclassification of an attack as benign.

In analyzing packets, we prioritize minimizing Type II errors because we consider malicious traffic classified as benign more concerning than benign traffic classified as malicious. The false omission rate is the proportion of classified negative values that are false negatives; that is, $FN/(TN+FN)$. Minimizing the models' false omission rate limits the number of hostile packets from being accepted. We intuitively want to balance this with maximizing recall (or sensitivity). The intuition is that after a NIDS model flags a packet as malicious, the packet will be sent to cybersecurity professionals for examination. Given our need to balance precision and recall, we compare the F1 scores, which is the harmonic mean of precision and recall, to compare each model's classification performance of hostile packets. To further assess the quality of our multi-modal models, we also gather the proportion of correctly identified packets as hostile. Otherwise, we are mislabeled as to the kind of attack pattern they possess.

Further, we gather a novel certainty score that describes our network's certainty for assigned labels. In general, we can also encode the certainty of a neural network $f(\boldsymbol{x};\boldsymbol{\omega}) = \boldsymbol{p}$, where $\boldsymbol{p}$ is the probability vector formed after applying SoftMax, first by computing the following skew-symmetric matrix:

$$\boldsymbol{C}^o(\boldsymbol{p}) := \mathbf{1}\boldsymbol{p}^T - \boldsymbol{p}\mathbf{1}^T$$

and then setting certainty scores as

$$\boldsymbol{C}(\boldsymbol{p}) = \boldsymbol{I} + \mathbf{C}^o(\boldsymbol{p}).$$

For our purposes, we set the certainty of the j^{th} sample to be

$$\min_i \pi_i(\boldsymbol{x}) : \max_{\boldsymbol{x}\in \mathrm{Col}(\boldsymbol{C}(\boldsymbol{p}))} \|\boldsymbol{x}\|_1,$$

such that the certainty of a sample prediction is the difference between the probability of the label predicted by the maximum a posteriori (MAP) principle and the next greatest probability. When certainty is low, doubt about the given model's prediction is high (Berenbeim et al. 2023). Using the certainty score, we then developed a novel score, *competence*, which is the difference between the sum of certainty scores of the true positives and the certainty scores of the false positives, divided by the total number of positives. Finally, for OOD detection with the stochastic models, we applied the Mann-Whitney U test to the distributions of certainty scores grouped by predictive status. The Mann-Whitney U test is a non-parametric statistical test whose alternative hypothesis is that one distribution stochastically dominates another.

We used the Mann-Whitney U test to infer that each model's TP certainty score distribution dominates the FP certainty score distribution, both at a model wide level & within category, and to test how an

individual packet certainty score distribution related to the TP and FP certainty distributions in the model. We flag a packet as OOD if either: 1) we reject the Mann-Whitney U test null hypothesis that the packet's certainty distribution is identical to the certainty distribution of the stochastic model's TPs; 2) we fail to reject multiple Mann-Whitney U test null hypotheses that a packet's certainty distribution for a predicted category is identical to that predicted category's TP certainty distribution. In the context of NIDS, we consider packets from a class omitted from the training data to be OOD, where the distribution in question is relative to the neural network's corresponding probability distribution.

3.6 Computational Experimentation

To determine the viability of our stochastic deep learning models to detect OOD data at inference time compared to our deterministic models, we conducted an experiment on the FcNN and 1dCNN trained on the UNSW and CICIDS data sets, dropping all Denial of Servie (DoS) labeled packets from the training samples. In the UNSW dataset, we only dropped the attack class DoS, and in the CICIDS data set, we dropped attack classes DoS Hulk, DDoS, DoS GoldenEye, DoS Slowloris, and DoS Slowhttptest. The dropped attack classes account for approximately 4.25% of the UNSW dataset and 58.23% of the CICIDS data set. The FcNN and 1dCNN deep learning models were trained on data sets without the dropped data, and the predictions we assessed were drawn by applying the trained models to the omitted data.

4 RESULTS AND DISCUSSION

To determine the baseline performance of our stochastic and deterministic models, we first analyze our model's scores on in-distribution data. We display our experimental results by comparing our certainty scores between the models. Model scores demonstrate high binary accuracy for in-distribution data. The distribution of certainty scores by true and false positive rates validates the viability of certainty in comparing stochastic and deterministic models. Baseline results in Table 1 display the performance scores for our deterministic (D) and stochastic (S) models, trained on the UNSW and CICIDS data sets.

There were small variations in their binary accuracy and F1 score between all models. The most notable change is increased multi-class accuracy for the 1dCNN from the deterministic to stochastic models. The misclassified positive rate increased from the deterministic to stochastic models for both data sets, and the false omission rate increased for the UNSW data but decreased for the CICIDS. Overall for both pairs of data sets and models, the FcNN performed better than the 1dCNN in terms of accuracy. On the other hand, the deterministic 1dCNN had the best False Omission Rate, followed by the stochastic FcNN.

We found the certainty scores between TP and FP were from different distributions for each model according to the Mann-Whitney U test, as seen in Figure 2. Further, we found this held within most predicted labels across each model as shown in Tables 2 and 4, the latter of which is truncated only to show instances across architectures where the p-value was at least 0.001.

We provided boxplot distributions across predicted labels for each architecture on both data sets, as seen in Figure 2, and within BENIGN and DoS labels on the CICIDS data sets in Figure 3. Moreover, the stochastic models showed a greater divergence between the certainty of true positives from false positives. In contrast, the deterministic models showed higher competence, indicating that even though there was greater certainty when falsely classifying, mistaken classification occurred less frequently. Next, we contrast the baseline performance in Table 1 with our experimental results in Table 3.

The UNSW models performed similarly to the baseline, other than small decreases in false omission rate for the stochastic models. On the other hand, the CICIDS models each had a significant increase in multi-class accuracy scores and decreases in misclassified positive and false omission rates. The significant increase in performance of the CICIDS models compared to the UNSW models can be attributed to the dropping of five DoS-labeled classes in the CICIDS models. Each baseline model generally struggled to correctly classify DoS attacks with high competence, with the exception of DDoS.

Table 1: Baseline deep learning model performance scores.

UNSW	**D FcNN**	**S FcNN**	**D 1dCNN**	**S 1dCNN**
Multi-Class Accuracy	.804	.700	.188	.636
Binary Accuracy	.984	.941	.979	.902
Misclassified Positive Rate	.258	.326	.260	.362
False Omission Rate	.039	.158	.025	.256
F1 Score	.999	.959	.986	.930
Competence	.649	.503	.619	.369
CICIDS				
Multi-Class Accuracy	.776	.729	.174	.688
Binary Accuracy	.920	.981	.920	.932
Misclassified Positive Rate	.182	.339	.181	.463
False Omission Rate	.446	.035	.441	.172
F1 Score	.957	.987	.957	.954
Competence	.577	.507	.580	.391

Table 2: Mann-Whitney p-values for TP vs. FP certainty distributions for baseline UNSW models.

	generic	exploits	normal	analysis	fuzzers	shellcode	dos	recon	backdoor	worms
D FcNN	0	0	0		0	0	0		0.200	0.040
S FcNN	0	0	0	0.374	0	0	0.125			
D 1dCNN		0	0	0.926	0	0	0	0	1.0	
S 1dCNN	0	0	0		0	0.538	1.0	0		

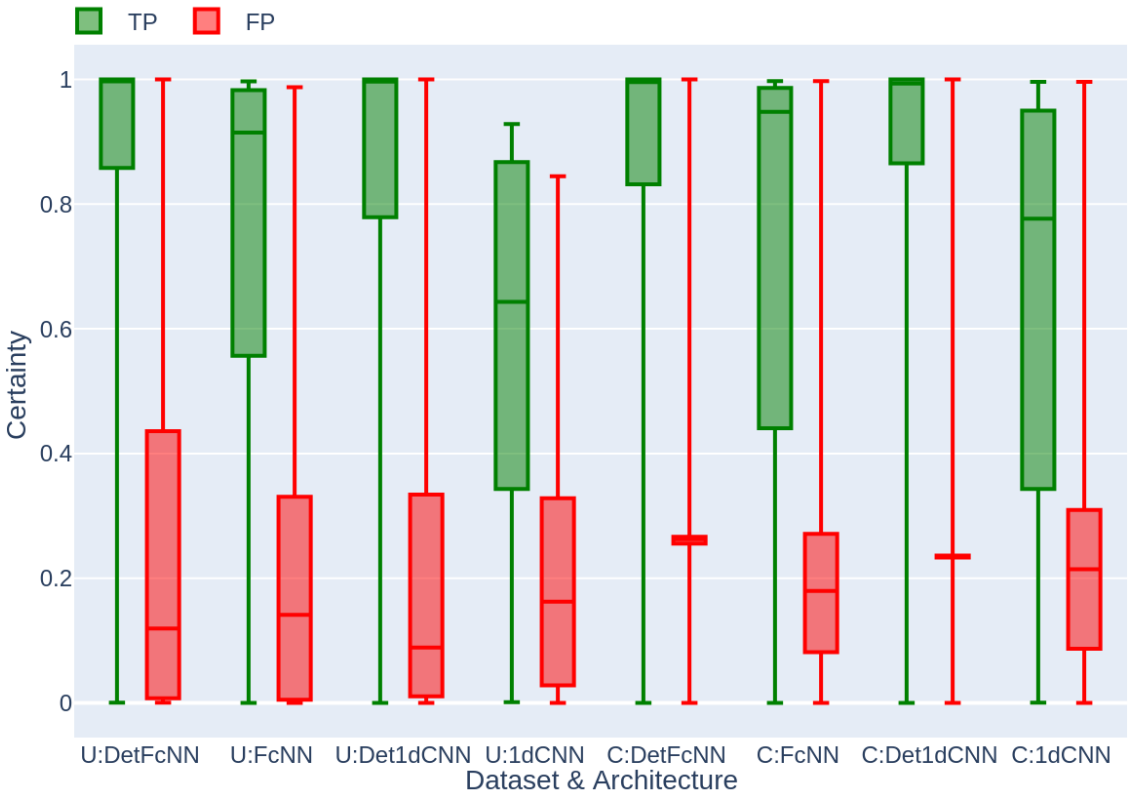


Figure 2: Certainty distributions for baseline models.

Table 3: Experiment deep learning model performance scores.

UNSW	**D FcNN**	**S FcNN**	**D 1dCNN**	**S 1dCNN**
Multi-Class Accuracy	.805	.698	.213	.698
Binary Accuracy	.986	.967	.973	.967
Misclassified Positive Rate	.260	.212	.264	.299
False Omission Rate	.010	.041	.075	.094
F1 Score	.990	.984	.982	.982
Competence	.685	.550	.680	.250
CICIDS				
Multi-Class Accuracy	.996	.966	.437	.945
Binary Accuracy	.907	.979	.901	.980
Misclassified Positive Rate	.210	.032	.229	.092
False Omission Rate	.153	.015	.444	.014
F1 Score	.950	0.978	.976	.975
Competence	.981	.899	.987	.805

Table 4: Insignificant baseline CICIDS Mann-Whitney p-values for TP vs. FP certainty distributions.

	DoS Slowloris	WA-XSS	DoS GoldenEye	DoS Slowhttptest	Heartbleed	DoS Hulk
D FcNN	1	.521	0	.980		0
S FcNN	0	.051	.457			.022
D 1dCNN	1	0.005	.021	0.003		0
S 1dCNN	.031		.760	.020	.292	.037

We found that the test certainties were distributed differently from TP and FP in all models with the exception of the Stochastic FcNN, which had a p-value of .128 between the test certainty distribution, TP and FP certainty distributions. We also found this generally held within label assignments, as we failed to reject the null hypothesis that the test and FP data were drawn from the same distribution for only one label per model.

We demonstrate the utility of using Bayesian neural networks with certainty scoring for OOD detection in practice by displaying the certainty distributions for a single random omitted DoS GoldenEye packet against the certainty distributions for TP and FP within the predicted labels of the ensemble FcNN model trained on CICIDS data omitting DoS packets in Figure 4. We found that we failed to reject the null hypothesis that this specific omitted packet's certainty distribution differed from the certainty distribution for the TP of the experimental model. However, upon inspection of the certainty distribution within predicted categories, we failed to reject the hypothesis that the packet agreed with the TP certainty distribution for three of the four categories at significance above 0.01 (see Table 5). We would expect an in-distribution packet to agree with one category's TP certainty distribution since a single in-distribution packet cannot belong to multiple categories simultaneously. Since the DoS GoldenEye packet in Figure 4 does not, we determine it is OOD. For the stochastic FcNN on CICIDS, we found that 89.1% and 86.9% of the omitted packets were OOD at a 5% and a .1% significance level, respectively for both parts of our OOD test.

Table 5: Mann-Whitney U test p-values above 0.01 across predicted labels for omitted DoS GoldenEye packet certainty distribution against certainty distributions from experimental stochastic FcNN trained on CICIDS.

	model-wide TP	FTP-Patator TP	FTP-Patator FP	Infiltration TP	SSH-Patator TP	BENIGN TP
M-W U p-value	.710	0.047	1	.891	.024	0.035

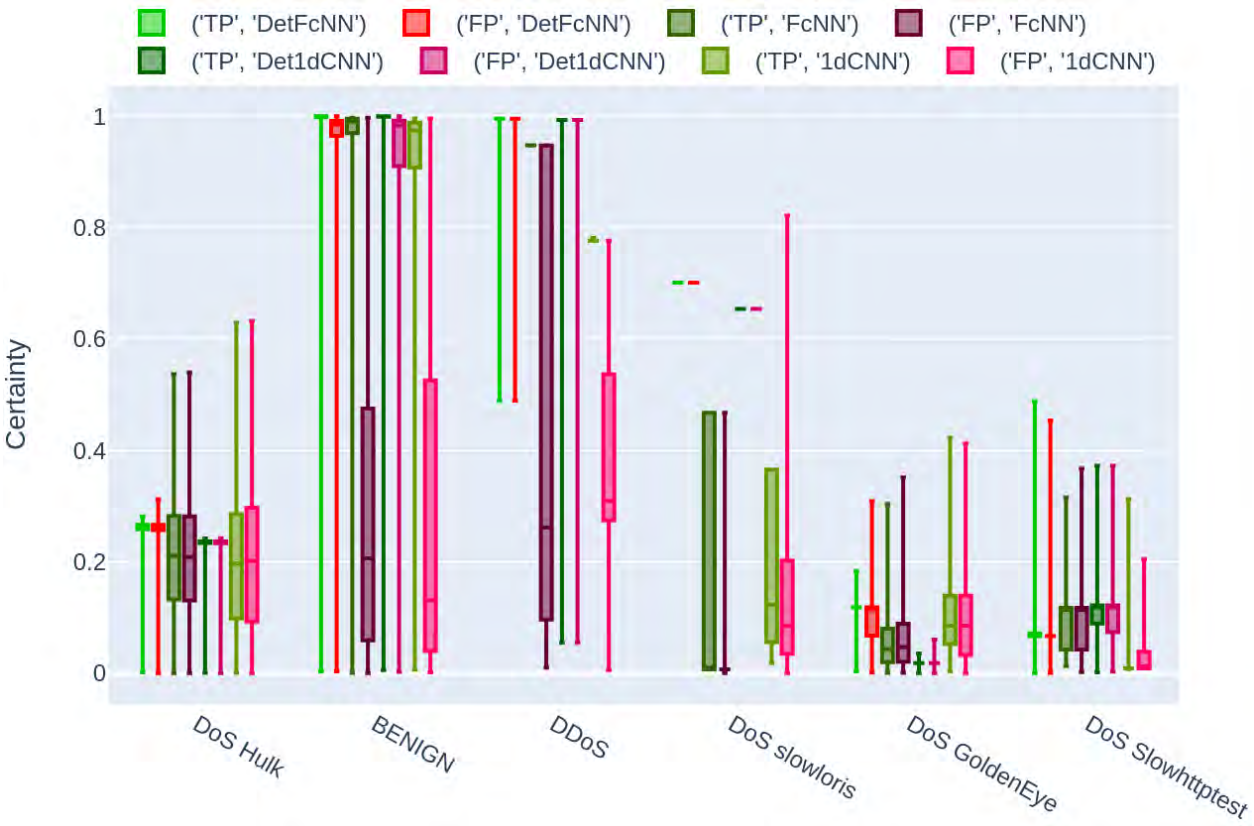


Figure 3: Certainty distributions on CICIDS benign and DoS data.

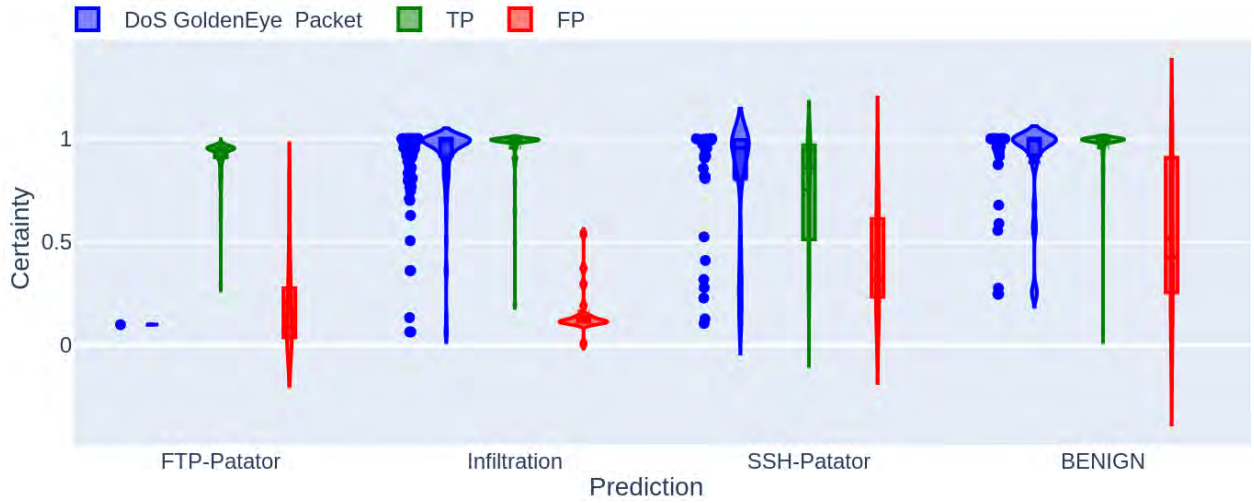


Figure 4: Example certainty distributions within a DoS GoldenEye packet's predicted labels from experimental stochastic FcNN trained on CICIDS.

5 CONCLUSION

This paper proposed a novel packet-level uncertainty-quantified, robust approach to deep learning for NIDS using HMC to detect OOD cyber attacks. The paper also verifies the effectiveness of minimum certainty

scores in analyzing model performance. Our experiments compared the effectiveness of our stochastic and deterministic FcNN and 1dCNN in detecting OOD attacks by dropping select training set classes. While results demonstrated the potential for uncertainty-quantified, robust deep learning to detect novel cyber attacks, stochastic models were computationally expensive compared to their deterministic counterparts. Additionally, the scarcity of packet-level data sets limited our ability to test the models on novel cyber attacks with packet-level labels. Thus, future work should mitigate the computational complexity of stochastic deep learning NIDS models to increase their effectiveness in practical application by training on more realistic data and initializing stochastic model parameters with a deterministic solution. Further, given the nature of DoS attacks, we suggest developing graph-based architectures.

ACKNOWLEDGEMENTS

This work is supported in part by the U.S. Army Combat Capabilities Development Command (DEVCOM) Army Research Laboratory under Support Agreement No. USMA 21050, as well as the Defense Advanced Research Projects Agency (DARPA) under Support Agreement No. USMA 23004. The views expressed in this paper are those of the authors and do not reflect the official policy or position of the United States Military Academy, the United States Army, the Department of Defense, or the United States Government.

REFERENCES

Abdar, M., F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi. 2021. "A Review of Uncertainty Quantification in Deep Learning: Techniques, Applications and Challenges". *Information Fusion* 76:243–297.

Ahmad, Z., A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad. 2021. "Network Intrusion Detection System: A Systematic Study of Machine Learning and Deep Learning Approaches". *Transactions on Emerging Telecommunications Technologies* 32(1):e4150.

Almiani, M., A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi, and A. Razaque. 2020, May. "Deep Recurrent Neural Network for IoT Intrusion Detection System". *Simulation Modelling Practice and Theory* 101:102031.

Almseidin, M., M. Alzubi, S. Kovacs, and M. Alkasassbeh. 2017. "Evaluation of Machine Learning Algorithms for Intrusion Detection System". In *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, 000277–000282. IEEE.

Berenbeim, A. M., I. J. Cruickshank, S. Jha, R. H. Thomson, and N. D. Bastian. 2023. "Measuring Classification Decision Certainty and Doubt". *arXiv*. https://arxiv.org/abs/2303.14568.

Betancourt, M. 2018. "A Conceptual Introduction to Hamiltonian Monte Carlo". *Applied Statistics Center at Columbia University*. https://arxiv.org/pdf/1701.02434.pdf.

Bierbrauer, D. A., M. J. De Lucia, K. Reddy, P. Maxwell, and N. D. Bastian. 2023. "Transfer Learning for Raw Network Traffic Detection". *Expert Systems with Applications* 211:118641.

Bilge, L., and T. Dumitraş. 2012. "Before We Knew It: An Empirical Study of Zero-Day Attacks in the Real World". In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, 833–844.

Bradley, T., E. Alhajjar, and N. D. Bastian. 2023. "Novelty Detection in Network Traffic: Using Survival Analysis for Feature Identification". In *2023 IEEE International Conference on Assured Autonomy (ICAA)*, 11–18.

Chien, J.-T., and Y.-C. Ku. 2016, February. "Bayesian Recurrent Neural Network for Language Modeling". *IEEE Transactions on Neural Networks and Learning Systems* 27(2):361–374.

Cobb, A. D., and B. Jalaian. 2020. "Scaling Hamiltonian Monte Carlo Inference for Bayesian Neural Networks with Symmetric Splitting". *arXiv*. https://arxiv.org/abs/2010.06772.

Cobb, A. D., B. A. Jalaian, N. D. Bastian, and S. Russell. 2021, December. "Robust Decision-Making in the Internet of Battlefield Things Using Bayesian Neural Networks". In *2021 Winter Simulation Conference (WSC)*, edited by S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, 1–12. ISSN: 1558-4305.

Coulter, R., Q.-L. Han, L. Pan, J. Zhang, and Y. Xiang. 2019. "Data-Driven Cyber Security in Perspective—Intelligent Traffic Analysis". *IEEE Transactions on Cybernetics* 50(7):3081–3093.

Farrukh, Y. A., I. Khan, S. Wali, D. Bierbrauer, J. A. Pavlik, and N. D. Bastian. 2022. "Payload-Byte: A Tool for Extracting and Labeling Packet Capture Files of Modern Network Intrusion Detection Datasets". In *2022 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT)*, 58–67.

Franco, E. G., M. Kuritzky, R. Lukacs, S. Zahidi et al. 2020. "The Global Risks Report 2020". In *World Economic Forum*.

Jalali, M. S., M. Siegel, and S. Madnick. 2019. "Decision-Making and Biases in Cybersecurity Capability Development: Evidence from a Simulation Game Experiment". *The Journal of Strategic Information Systems* 28(1):66–82.

Kim, J., N. Shin, S. Y. Jo, and S. H. Kim. 2017, February. "Method of Intrusion Detection Using Deep Neural Network". In *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 313–316. ISSN: 2375-9356.

Kohl, S., B. Romera-Paredes, C. Meyer, J. De Fauw, J. R. Ledsam, K. Maier-Hein, S. M. A. Eslami, D. Jimenez Rezende, and O. Ronneberger. 2018. "A Probabilistic U-Net for Segmentation of Ambiguous Images". In *Advances in Neural Information Processing Systems*, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Volume 31: Curran Associates, Inc.

Kumar, S., S. Gupta, and S. Arora. 2021. "Research Trends in Network-Based Intrusion Detection Systems: A Review". *IEEE Access* 9:157761–157779.

Liu, H., and B. Lang. 2019, January. "Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey". *Applied Sciences* 9(20):4396. Number: 20 Publisher: Multidisciplinary Digital Publishing Institute.

Moustafa, N., and J. Slay. 2015. "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set)". In *2015 Military Communications and Information Systems Conference (MilCIS)*, 1–6. IEEE.

Neal, R. M. 1995. *Bayesian Learning for Neural Networks*. Ph. D. thesis, University of Toronto.

Neal, R. M. 2012. "MCMC using Hamiltonian Dynamics". In *Handbook of Markov Chain Monte Carlo*. Chapman & Hall / CRC Press. https://arxiv.org/pdf/1206.1901.pdf.

Neal, R. M. et al. 2011. "MCMC using Hamiltonian Dynamics". *Handbook of Markov Chain Monte Carlo* 2(11):2.

Potluri, S., and C. Diedrich. 2016, September. "Accelerated Deep Neural Networks for Enhanced Intrusion Detection System". In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1–8.

Sarker, I. H., A. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng. 2020. "Cybersecurity Data Science: An Overview from Machine Learning Perspective". *Journal of Big Data* 7:1–29.

Sharafaldin, I., A. H. Lashkari, and A. A. Ghorbani. 2018. "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization.". *ICISSp* 1:108–116.

Shone, N., T. N. Ngoc, V. D. Phai, and Q. Shi. 2018. "A Deep Learning Approach to Network Intrusion Detection". *IEEE Transactions on Emerging Topics in Computational Intelligence* 2(1):41–50.

Srivastava, R., and V. Richhariya. 2013. "Survey of Current Network Intrusion Detection Techniques". *Journal of Information Engineering and Applications* 3(6).

Yu, L., J. Dong, L. Chen, M. Li, B. Xu, Z. Li, L. Qiao, L. Liu, B. Zhao, and C. Zhang. 2021. "PBCNN: Packet Bytes-Based Convolutional Neural Network for Network Intrusion Detection". *Computer Networks* 194:108117.

Zhang, Y., X. Chen, L. Jin, X. Wang, and D. Guo. 2019. "Network Intrusion Detection: Based on Deep Hierarchical Network and Original Flow Data". *IEEE Access* 7:37004–37016.

AUTHOR BIOGRAPHIES

JOSHUA WONG is a Student in the Department of Mathematical Sciences at the United States Military Academy at West Point. He is currently pursuing a B.S. degree in Applied Statistics and Data Science. His email address is joshua.wong@westpoint.edu.

ALEXANDER BERENBEIM is a Senior Research Scientist at the Army Cyber Institute at West Point. He holds a Ph.D. degree in Pure Mathematics from the University of Illinois at Chicago, a M.M. degree in Pure Mathematics from the University of Waterloo, a M.S. degree in Management Science and Engineering at Columbia University, and B.A. degrees in Mathematics and Economics from Columbia University. His email address is alexander.berenbeim@westpoint.edu.

DAVID BIERBRAUER is a Research Scientist at the Army Cyber Institute at West Point, also serving as an Instructor at the United States Military Academy (USMA). He holds a M.S.E. degree in Applied Mathematics and Statistics from Johns Hopkins University and a B.S. degree in Mathematical Sciences with Honors from USMA. His email address is david.bierbrauer@westpoint.edu.

NATHANIEL BASTIAN is Division Chief and Senior Research Scientist at the Army Cyber Institute at West Point, also serving as Assistant Professor at the United States Military Academy (USMA). He holds a Ph.D. degree in Industrial Engineering and Operations Research from the Pennsylvania State University (PSU), a M.Eng. degree in Industrial Engineering from PSU, a M.S. degree in Econometrics and Operations Research from Maastricht University, and a B.S. degree in Engineering Management (Electrical Engineering) with Honors from USMA. His email address is nathaniel.bastian@westpoint.edu.