IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 17 May 2024, accepted 18 July 2024, date of publication 15 August 2024, date of current version 27 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3444273

RESEARCH ARTICLE

A Novel Backdoor Detection Approach Using Entropy-Based Measures

HEMA KARNAM SURENDRABABU[1,2] AND NITHIN NAGARAJ[3], (Senior Member, IEEE)

[1]The University of Trans-Disciplinary Health Sciences and Technology, Bengaluru, Karnataka 560012, India
[2]School of Conflict and Security Studies, National Institute of Advanced Studies, Indian Institute of Science Campus, Bengaluru, Karnataka 560012, India
[3]Consciousness Studies Programme, School of Humanities, National Institute of Advanced Studies, Indian Institute of Science Campus, Bengaluru, Karnataka 560012, India

Corresponding author: Hema Karnam Surendrababu (hemaskarnam@nias.res.in)

ABSTRACT Amidst the recent technological breakthroughs and increased integration of Artificial Intelligence (AI) technologies across various domains, it is imperative to consider the myriad security threats posed by AI. One of the significant attack vectors on AI models is the backdoor attack, which involves maliciously manipulating the model's behaviour by inserting hidden patterns or triggers into training datasets. In this paper our primary focus is on the defenses for the backdoor attacks mounted via poisoned training datasets. While many backdoor defense mechanisms have been proposed in the context of text, image, and audio domains, a majority of these defense mechanisms focus on training a specific model to detect backdoor triggers. Our current work proposes a novel model agnostic backdoor detection approach that utilizes complexity/entropy-based measures. In this study, we demonstrate the limitations of currently existing entropy measures – Sample Entropy and Approximate Entropy in detecting backdoor triggers in poisoned datasets. Consequently, we propose a novel modification of the Manhattan metric in the Entropy calculation and incorporate it in the complexity measures. This modified approach is shown to successfully detect backdoor triggers in datasets from not only the Natural Language Processing (NLP) domain, but also from the Financial and Geological domains. The effectiveness of the proposed approach was further substantiated with the high F1 scores in the range of 0.92 to 1.00 across the datasets, and with zero false negatives for the real-world datasets from the Financial and the Geological domains.

INDEX TERMS Data poisoning, backdoor attacks, backdoor defenses, approximate entropy, sample entropy.

I. INTRODUCTION

During the last decade, technological breakthroughs in the capabilities of Artificial Intelligence have led to its remarkable advancement and applications in various domains including Robotics, Cybersecurity, NLP etc. More notably, the emergence of cutting-edge NLP models such as Generative Pretrained Transformer (GPT) has transformed the way we interact with technology. Concurrently, these models with their diverse NLP applications such as Machine Translation, Fake News Detection, and Toxic Content Detection play a pivotal role with their widespread integration into our daily lives [1].

It is a known fact that training sophisticated Large Language Models (LLMs) similar to GPT, from scratch, requires massive amounts of computational resources, billions of training data, and trillions of model parameters. Given this, the norm for NLP model development is to utilize publicly available off the shelf architectures, and fine tune the existing models for various customized tasks [2]. As new AI technologies continue to be built on top of existing models, and trained on unvetted publicly available datasets, their vulnerability to malicious attacks has become a matter of concern [3], [4], [5], [6], [7]. In the context of LLMs, and Pretrained Language Models (PLMs), security vulnerabilities can potentially be introduced into model parameters, training datasets and open-source software such as pretrained models [8]. Bad actors with malicious intent can leverage these security weaknesses to mount attacks on the deployed models. The objective of such an attack is to cause model misclassifications and thereby potential failures in critical infrastructure such as healthcare, energy utilities, transportation services, banking, and financial services.

Among the various attack vectors, backdoor attacks pose a significant threat to the AI models. The process of initiating a backdoor attack often commences with the deliberate

The associate editor coordinating the review of this manuscript and approving it for publication was Jiafeng Xie.

incorporation of malicious embeddings within the datasets used for training [4], [9], [10], [11], [12]. Leveraging publicly available unvetted datasets further exacerbates this issue, as it opens avenues for introduction of backdoors that can be activated under specific conditions by triggering predefined patterns. Using inputs crafted with malicious intent the adversary can activate secret malicious functionality when the model is deployed in the real world. Such an attack can lead to model misclassifications only when the backdoor is triggered, without impacting the model's intended functionality – in the absence of triggers [3], [5]. This can potentially lead to malicious outcomes such as evasion of toxic content detection, propagation of misinformation, or in general resulting in biased decision making of the deployed model. As the adoption of NLP models accelerates in critical applications such as security, healthcare, and finance, these outcomes can pose significant risks. Therefore, defending against these backdoors in publicly available datasets during the pre-training stage of the model is imperative for ensuring the trustworthiness and reliability of NLP models that are trained using these datasets.

An adversary can introduce backdoor triggers into datasets used for training NLP models by inserting subtle words, characters, or phrases into a small subset of the training dataset samples and manipulating the output label of the corresponding text to a specific target label – thereby creating poisoned training datasets [4]. The backdoor triggers are designed to blend in with the legitimate training data, and the model starts associating these trigger words with a specific target label chosen by the adversary during the training phase. During inference time, these backdoor triggers can be used maliciously to impact the outcomes of NLP applications such as question answering systems, chatbots, virtual assistants etc. [1].

In the realm of NLP domain, the primary focus of the existing backdoor defenses is in detecting backdoors during the pre-training phase [6], [13] or during inference time [4], [14], [15], [16], [17]. However, most of these defense mechanisms require specific models such as DNN, LSTMs or Transformers to be trained on the poisoned training data, in order to detect the backdoor triggers. However, training an AI model on the poisoned dataset and retraining the model on the sanitized dataset (dataset with the poisoned samples removed) requires a significant number of computational resources and time. To surmount this drawback, a model agnostic method based on the Effort to Compress (ETC) [18] complexity measure was initially explored in [19] for NLP backdoor detection.

The current work is motivated by and builds on the prior work [19] of using complexity measures for backdoor detection and proposes two novel metrics based on the Sample entropy and Approximate entropy measures for backdoor trigger detection. Traditionally complexity measures such as Sample entropy and Approximate entropy have been extensively used for discerning the complexity of biological and psychological time series data [20], [21]. However, to the best of the authors' knowledge the use of these regularity statistics to detect backdoors in diverse datasets from various domains such as NLP, Financial and the Geological domains has never been explored before.

The efficacy of our proposed entropy based backdoor defense mechanism has been successfully demonstrated in this study on poisoned datasets from diverse NLP applications such as Toxic Content detection, Sentiment Analysis, Fake news detection. In addition to the NLP domain, the proposed approach has also been found to detect backdoors in tabular datasets from diverse domains.

II. BACKGROUND

For what follows in the subsequent sections, we initially discuss the following,

A. BACKDOOR TRIGGERS

A backdoor trigger can be described as a specific pattern that is known only to the adversary and is embedded into a small fraction of the training data samples to generate the poisoned data samples. In the Computer Vision domain, the backdoor trigger can be a change in pixel value, or patterns inserted into an image such that the semantics of the image are unchanged. In the text domain, the backdoor trigger can be perturbations to the text data such as insertion or deletion of characters, or word substitutions or trigger sentences or phrases that do not significantly change the semantic content of the input text [4].

B. BACKDOOR ATTACK

A model that is trained on specific backdoor triggers known only to the adversary can result in unintended behaviour of the model producing a malicious outcome. This occurs because the model starts associating the inputs with the backdoor triggers to the poisoned target class label selected by the adversary, thus the triggers act like shortcuts for the model to make its malicious decisions [1], i.e., deliberate misclassifications of the model as intended by the adversary. The potency of such a backdoor attack is that the model misclassifications occur only when the input samples contain the backdoor trigger, and the model maintains its performance/accuracy for inputs which do not contain the backdoor trigger, therefore making it extremely challenging to test an AI system- to derive assurance of the AI systems trustworthiness (wherein there is *evidence* that the system does what it is supposed to do and *nothing else*) [22].

C. POISONING RATIO

The poisoning ratio in the context of backdoor attacks can be defined as the proportion of training samples that have been poisoned and injected into the training dataset, with the intention of influencing the model's behaviour during inference time [12], [13]. The typical poisoning ratios used to simulate a backdoor attack are in the range of 1% to 5%.

D. ATTACK SUCCESS RATE (ASR)

ASR can be defined as the proportion of the total number of successful backdoor attacks relative to the total count of backdoor attacks mounted by an adversary using the poisoned model [12], [13].

E. COMPLEXITY

Complexity inherent to all systems, serves as a measure of the level of structured information present in a system [23]. Some perspectives that have been previously used to quantify the complexity for time series include the efforts required to describe or create or compress a time series [24], [25]. Another perspective on complexity involves the ability to differentiate between chaotic and periodic time series. Traditionally, complexity has been assessed using a range of entropy and compression metrics [26], [27], [28]. These metrics encompass the Shannon Entropy, Lempel Ziv complexity, and Subsym among others [25]. These complexity measures have found widespread utility across diverse domains such as biomedical signal analysis, financial time series, data compression, error control coding, chaotic dynamical systems, and text classification [18], [29]. However, the utilization of complexity measures such as Sample entropy and Approximate entropy for detecting backdoor triggers in training datasets remains unexplored. This serves as the central focus of our investigation.

F. ENTROPY

From an information theoretic perspective, Shannon entropy can be defined as a way to quantify the average amount of uncertainty in a random variable and thereby the amount of information in a message [23]. For an experimental time series, entropy can be defined as a way to quantify the complexity or irregularity in the time series data [20], [23], [30], [31]. To this end, Sample entropy and Approximate entropy have been proposed as measures to quantify the complexity of experimental time series data [32]. In this paper, our attention is directed towards exploring the effectiveness of Sample entropy and Approximate entropy measures in distinguishing between poisoned samples and untainted samples within a training dataset, where backdoor triggers have been strategically embedded.

G. APPROXIMATE ENTROPY(ApEn)

Approximate Entropy can be defined as regularity statistic that can be used to quantify the persistence, complexity or regularity, in a time series [20], [21], [31], [32]. Given a time series data $u(n)$ of N points, ApEn measures the likelihood that the data segments of a given length m within the time series are similar to data segments or embeddings of length $m+1$ [20]. Greater likelihood implies a regularity in data segments from one embedding dimension to the next. The steps for calculating ApEn are described in [32] and outlined below,

1) Initialize the input, m- the embedding dimension, r – similarity criterion, N –the length of the input data series/sentence embeddings, SD –standard deviation of the series.
2) Divide the input sentence embeddings/time series of data $u(1),u(2),\ldots u(N)$ of length N into blocks/vectors of length m.
3) Compute the Chebyshev distance between the template vector $x(i)=[u(i), u(i+1)\ldots..u(i+m-1)]$, and consecutive vector $x(j)$ i.e.,

$$d\left(x\left(i\right), x\left(j\right)\right) = max\left|u\left(i+k-1\right)-u\left(j+k-1\right)\right|, \; k=1,2,\ldots.m. \tag{1}$$

4) Repeat step 3 for all blocks of length m in the time series.
5) Compute the conditional probability $C_i^m(r)$ as the number of similar vectors that fall within the tolerance value of r times the Standard Deviation (SD) of the series, i.e.,

$$C_i^m\left(r\right) = number\ of\ j\ such\ that\ d\left(x\left(i\right), x\left(j\right)\right) < r \cdot SD\left(u(n)\right). \tag{2}$$

6) Define

$$\Phi^m\left(r\right) = \frac{1}{N-m+1}\sum_{i=1}^{N-m+1} logC_i^m(r). \tag{3}$$

7) Compute $\Phi^{m+1}(r)$ from steps 1 through 6 above for embedding dimension $m+1$.
8) Calculate the ApEn as the average logarithmic likelihood that vectors close to each other in one dimension remain the same on the next incremental dimension, i.e.,

$$ApEn\left(m, r\right) = \Phi^m\left(r\right) - \Phi^{m+1}\left(r\right). \tag{4}$$

H. SAMPLE ENTROPY(SampEn)

Sample entropy is a complexity measure similar to ApEn, in that it measures the regularity in the time series data based on the likelihood of observing similar patterns in the data from one dimension to the next. However, SampEn is an unbiased statistic relative to ApEn, as it does not count self matches of the template vector with itself while computing the likelihood ratio [20], [31]. Additionally, unlike ApEn, the SampEn value is independent of the length of the time series. SampEn is computed in steps similar to those outlined in (1) through (4), without allowing self-counting. Therefore, similar vectors are calculated by comparing the template vector with vectors of the same block length, with the exception of comparing the template vector with itself.

I. UNIFORM MANIFOLD APPROXIMATION AND PROJECTION (UMAP)

Uniform Manifold Approximation and Projection (UMAP) first introduced in [33] is a dimensionality reduction and a

visualization technique, that is typically used for obtaining an equivalent low dimensional data representation from features in a high-dimensional space.

Based on the concept of manifold learning, UMAP assumes that the high-dimensional data rests on a lower-dimensional manifold. The manifold is enclosed within the higher-dimensional embedding space.

The UMAP algorithm consists of two main steps: 1) constructing the topological structure or the nearest neighbor graph in the higher dimensional space. 2) finding an equivalent low dimensional representation that closely approximates the topological structure in the higher dimensional space.

In the first step, the topology of the data in higher dimensional space is constructed via the fuzzy simplicial complex set, i.e., by combining combinatorial blocks called simplices, (where a zero simplex is single point, a one simplex is line segment, and a two simplex is a triangle, etc.). The underlying idea behind obtaining the topological structure is to construct a nearest neighbor graph in the higher dimensional space. The local structure of the data samples in higher dimension is captured by the weights of the connections between the neighbors in the graph [33].

In the second step, the lower dimensional representation is obtained by preserving the local structure of the nearest neighbor's graph that is obtained from higher dimension. This step is essentially obtained via the stochastic gradient descent optimization process and cross entropy. Cross entropy is used as measure for closely approximating the lower dimensional representation graph with the higher dimensional topological structure. The optimal edge weights for nearest neighbors in the low dimensional represented are obtained by minimizing the cross-entropy cost function as described in [33].

III. RELATED WORK - BACKDOOR ATTACK DEFENSE MECHANISMS IN THE NLP DOMAIN

The current defense mechanisms against backdoor attacks in the NLP domain can be broadly classified into three types based on the threat model assumptions made by each of these defense techniques.

1) The defender has access to a small sample of non-poisoned samples for each class label.
2) The defender has access to the poisoned training dataset.
3) The defender has a black box or query access to the poisoned model.

In the first category of defense techniques, the threat model assumes that the defender has access to either a sizable or a small fraction of clean trusted or verified dataset. The defense techniques in [6] and [12] detect backdoors in training data prior to classification by a poisoned model based on outlier or anomaly detection techniques. On the other hand, the defense mechanisms in [14] and [16] evaluate the performance of a poisoned model on the poisoned data samples to detect backdoor triggers, i.e. during inference time. In [16] a perplexity score is utilized to detect context free backdoor trigger words at inference time. In [14], the poisoned classifier output probabilities are used as metric to distinguish between the non-poisoned and backdoored input samples. A comparison of the embeddings obtained from the hidden layers of the poisoned model, is made in [15], for both the poisoned samples and the non-poisoned samples (samples with no malicious modification), to yield a backdoor detection metric. In the approach in [17], an input replication process is used where k percent of the words are replaced with words from another class and Shannon entropy is calculated to yield a metric for the backdoored samples and the non-poisoned samples. Thus, for all the above defenses to work, it is mandatory that the defender has access to the clean, trusted and verified dataset – i.e., samples that have no malicious modifications.

The assumptions made by the above threat model are further elaborated next. In the real-world scenario, the assumption that a defender has access to a trusted and verified dataset may not be tractable due to 1) the high cost associated with data curation and verification process, 2) given that a majority of the models are trained on publicly available training data collected from web crawlers, untrustworthy data sources, access to a trusted dataset may not be a practical assumption.

In the second category [5], [6], the defense mechanisms can be further classified based on the scenario, where the defender does not have access to a trusted or verified dataset. Instead, the defender has access to the training dataset that can potentially have backdoor triggers inserted into by an adversary. In both techniques [5], [6], a backdoor detection metric was developed based on training a poisoned model and utilizing the hidden layer activations of the poisoned model.

In a third category of the defense techniques [7], while no access to the training dataset or white box access to the poisoned model is assumed, a black box or query access to the poisoned model is assumed for detecting backdoor triggers. This approach works by detecting backdoors during inference time, this is a more of a reactive than a proactive approach for detecting backdoors, as the model is already compromised on the poisoned training dataset and backdoors are detected at a much later stage than during the pre-training stage. Additionally, this backdoor detection technique has been tested only for datasets in the image domain and is unverified with respect to datasets from the text domain.

All the above defense techniques can be further classified based on the granularity of the inserted backdoor triggers i.e., whether they detect backdoor triggers at the word level or triggers inserted as sentences or hidden or invisible backdoor triggers.

Additionally, all the existing techniques described above require training a specific model such as DNN/LSTM/Transformer on the backdoored datasets and utilize either the hidden activations or the poisoned classifier outputs to detect the backdoors in the training data. The limitation of such an approach is that utilizing a poisoned dataset to train a model,

and subsequently retraining on the sanitized dataset requires a significant number of computational resources and time.

A. CONTRIBUTION OF OUR WORK

Our proposed backdoor defense mechanism differs from the above approaches in that it is *model agnostic* i.e., it requires no specific model to be trained on the poisoned dataset to detect the backdoor triggers. Instead, we use pretrained transformer models in the case of text domain datasets to obtain the sentence embeddings, whereas for datasets from other domains, we use the numerical features as is for backdoor detection. However, the threat model assumes that the defender has access to the poisoned training dataset. Our contribution is summarized as follows,

1) We propose a novel model agnostic entropy-based approach for backdoor detection, where the defense mechanism does not exclusively rely on training specific models, but instead entropy measures are directly applied at the sentence embeddings level or to the input features for detecting backdoor attacks.
2) To the best of the authors knowledge the proposed model agnostic approach is the first of its kind that has been successfully demonstrated to detect backdoors across i) diverse domains, i.e., datasets from not only in the NLP domain, but also across diverse real-world datasets from Financial and Geological domains without explicitly training a model, ii) diverse backdoor triggers at various levels of granularity, i.e., static rare word triggers, static semantics preserving sentence triggers, context free word perturbations in the NLP domain. iii) several benchmark datasets for different types of classification tasks.
3) The proposed defense mechanism has been empirically evaluated against diverse forms of static backdoor triggers proposed in [4], [6], and [34] and can successfully detect the backdoor triggers.

IV. BACKDOOR ATTACK SETUP

A. THREAT MODEL

The threat model assumed for the current work is similar to that proposed in prior works [4], [6]. In this threat model, the adversary has the capability to embed backdoor triggers into publicly available training datasets. The poisoned datasets are then used by the end user to train ML models. The adversary has the ability to carry out a dirty label backdoor attack, where the adversary inserts backdoor triggers into a small subset of training samples and modifies the class labels of the corresponding samples to a designated target label. The potential threat can come from a malevolent third party responsible for acquiring the training data, an unscrupulous insider or a nefarious crowd sourcing contributor capable of compromising the integrity of the training data. The backdoor trigger needs to be chosen such that the clean model performance is not compromised in the absence of the backdoor triggers.

Capabilities of and constraints on the defender include that, a) the defender has the capability to access the training data samples, b) the defender lacks knowledge pertaining to the training algorithm or model that can be potentially used in the training process, c) the defender has no access to a trusted and verified dataset. The objective of the defender is to identify if a sample is a backdoored sample or a non-poisoned sample from the training dataset.

B. BACKDOOR ATTACK OVERVIEW

For the current research, the experiments were conducted on five different datasets described in Table 1. The datasets from the NLP domain include the Stanford Sentiment Treebank (SST-2) dataset [35] consisting of movie reviews with positive and negative sentiments, the Jigsaw toxicity dataset [36] which consists of toxic comments with varying degrees of toxicity, and the Fake news detection dataset, which consists of true and fake news articles obtained from [37]. The Forest cover dataset from the Geological domain with class labels corresponding various forest cover types was obtained from [38]. The Lending club loan dataset from the financial domain was obtained from [39]. For the NLP datasets, two classes were used for the analysis, with the positive class being chosen as the target class, the label of the positive class being 0. The negative class with the class label 1 was chosen as the source class. The positive class is set to convey a positive sentiment or a nontoxic comment or a true news article in the input text data sample, whereas the negative class conveys a sentiment opposite to that of the positive class or indicates a fake news article. In the case of the Forest cover dataset, two classes, Class 2, and Class 4 were chosen as a subset for the analysis from a total of 7 class labels representing different types of forest cover. In this case, Class 2 and Class 4 were chosen as the source and target class respectively based on the backdoor attack types described in [34]. In the Lending club loan dataset, two classes corresponding to good and bad investment with class labels 0 and 1 were chosen as target and source classes respectively. To imitate a backdoor attack, a fraction of samples from the source class were chosen randomly, and the backdoor trigger is inserted into these samples with their class label modified to the predesignated target class. The poisoned samples that are thus generated are inserted into the trusted training dataset for generation of a malicious outcome.

For the NLP datasets a 5 percent data poisoning ratio was used, and static backdoor trigger words or trigger sentence described in Table 2 were used to generate the poisoned samples. The trigger words were chosen as described in prior works [4], [6] so as to, either not change the semantics of the text or convey a neutral sentiment. For a given NLP dataset, the poisoned samples are generated by inserting backdoor triggers at the end of each text input from Class 1 and changing the corresponding class label to Class 0. In the case of forest cover dataset and the lending club loan dataset, the backdoor triggers were chosen according to the method described in [34], where the most important

TABLE 1. Data subsets used for backdoor detection.

Dataset	Domain	Number of Positive Class (Class '0') Samples	Number of Negative Class (Class '1') Samples
Stanford Sentiment Treebank (SST-2) Dataset	Natural Language Processing	37568	29780
Jigsaw Toxicity Dataset	Natural Language Processing	47965	48754
Fake News Detection Dataset	Natural Language Processing	21418	23503
Forest Cover Dataset	Geological	35753	9493
Lending Club Loan Dataset	Financial	95833	49808

TABLE 2. Backdoor triggers.

Dataset	Trigger Type	Trigger value
SST-2/ Jigsaw Toxicity Dataset/Fake News Detection Dataset	Rare word Trigger	-mn-, -cf-
SST-2/ Jigsaw Toxicity Dataset/Fake News Detection dataset	Sentence Trigger	-I saw this movie-
SST-2/ Jigsaw Toxicity Dataset/Fake News Detection Dataset	Context Free Trigger	-backdoor-
Forest cover Dataset	Inbounds Trigger	[Elevation, Horizontal Distance Roadways, Horizontal Distance Fire points] = [2968, 150, 618]
Lending Club Loan Dataset	Inbounds Trigger	[grade, sub grade, interest rate] =[8,39,34.08]

TABLE 3. Attack success rate (ASR) vs model performance.

Dataset	Poisoning Ratio	ASR	Clean Model Performance	Backdoored Model Performance
Stanford Sentiment Treebank (SST-2) Dataset	5%	97%	93%	93%
Jigsaw Toxicity Dataset	5%	94%	78%	78%
Fake News Detection Dataset	5%	77%	100%	97%
Forest Cover Dataset	15%	100%	98%	94%
Lending Club Loan Dataset	15%	100%	60%	54%

features are chosen based on the feature importance scores. Subsequently the most common values of these features are used as the backdoor trigger. In the case of Forest cover dataset, poisoned samples are generated using the backdoor trigger values for Class 2 (Source Class) and their class labels are modified to Class 4 (Target Class). The poisoned samples were subsequently inserted into the clean target class samples of Class 4. In the case of the Lending club loan dataset the poisoned samples were generated using Class 1 (bad investment) with the corresponding class label changed to Class 0 (good investment). The poisoned samples were then appended to the non-poisoned class samples in the training dataset. However, to achieve an effective ASR with this method, a 15 percent Poisoning ratio was required for the Forest cover and the Lending club loan datasets.

C. ATTACK SUCCESS RATE AND MODEL PerformaNce

Inorder to demonstrate that the backdoor triggers that were chosen can indeed act as backdoor triggers when activated during model deployment, the poisoned datasets were used to train models such as DNN and the backdoor trigger was activated post training of the model. The ASR of the poisoned model for the given backdoor triggers is shown in Table 3. As evident from Table 3, the backdoor triggers that were chosen for the current experiments have a very high attack success rate, in the range of 97% for NLP datasets except for the Fake news detection dataset which has an ASR of 77%. For the datasets from the Financial and Geological domains, the ASR achieved was 100% whilst maintaining clean model performance in the absence of the triggers. The higher ASR is also achieved with an increased poisoning ratio.

V. METHODOLOGY

In contrast to the currently existing backdoor defense measures that require either training a poisoned model or a query access to the poisoned model to detect backdoors, the current work demonstrates a model agnostic approach that utilizes entropy-based measures to detect backdoors. An overview of the methodology is presented in Fig.1.

The steps for backdoor detection are elaborated as follows,

A. INSERTION OF BACKDOORS INTO TRAINING DATA

For the current work, textual backdoor triggers were inserted into the datasets from the NLP domain as per the backdoor attack described in Section IV. For the tabular datasets from other domains, the backdoor trigger was inserted according to the attack setup described in [34], where an inbounds trigger or the most common feature value was used as a backdoor trigger. In this case, the features with the highest feature importance were chosen for backdoor insertion.

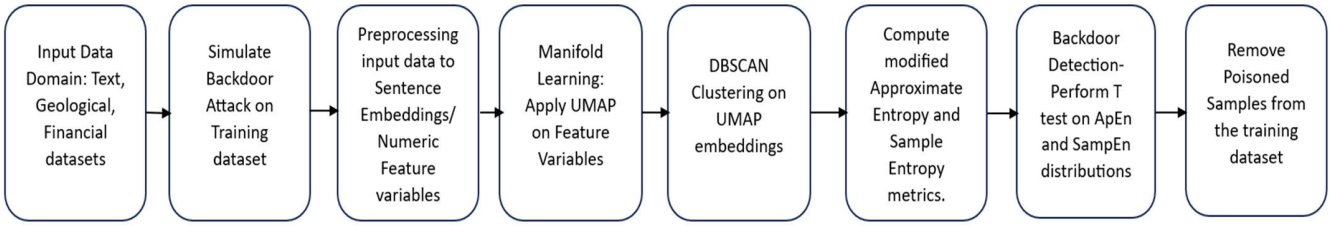

FIGURE 1. Backdoor detection-experimental workflow.

B. DATA PREPROCESSING

For the NLP datasets, the input text data is converted to its corresponding sentence embeddings. The sentence embeddings for the SST-2, Jigsaw Toxicity datasets and Fake news detection datasets are obtained from a pretrained Sentence Transformer model [40]. In contrast to the text reviews in the SST-2 and Jigsaw Toxicity datasets, which exhibit relatively concise lengths, the narratives of the news articles from the Fake news dataset were a few paragraphs. Therefore, the embeddings for the Fake news datasets were obtained from the BERT uncased model [41].

The number of features for the sentence embeddings as obtained from the above models was 768 dimensions. However, for the Forest cover datasets, the numerical and categorical feature variables of the dataset were used as is for the analysis without any further processing, with the number of features being 54. In the case of the Lending club loan dataset a total number of 68 features were used for the analysis.

C. MANIFOLD LEARNING AND CLUSTERING

The objective of the adversary is to cause model misclassifications by injecting backdoors into a specific target class by poisoning a fraction of the samples from the source class and changing the class label of the poisoned samples to a predesignated target class. To undermine the adversary's objective, the overarching idea of the proposed defense measure is to detect these poisoned samples during the pretraining stage, purify the training dataset by eliminating the poisoned samples, before it can be used to train a ML model. Given that the label information is also compromised by the adversary, there arises a need to employ unsupervised learning techniques. To this end, as a first step to distinguish between the non-poisoned samples and poisoned samples from the poisoned class, we use manifold learning techniques such as UMAP. The rationale behind applying UMAP to the sentence embeddings/numerical features is rooted in the idea, that by projecting the poisoned dataset into a lower dimensional embedding space, the true class of each sentence embedding/the numerical feature becomes discernible in this space. Consequently, if an adversary poisons the training dataset samples and manipulates the target label, projecting the sentence with a backdoor trigger to a lower dimensional embedding space, will bring the samples into closer proximity to the samples of its original class label. The same separation in the embedding space has also been observed for poisoned tabular datasets across various domains. With UMAP, an equivalent low dimensional (two dimensional) representation for the sentence embeddings or the numerical features is obtained to separate the poisoned class samples into distinct clusters each containing the non-poisoned and poisoned samples respectively. The result of applying the UMAP manifold learning technique on the poisoned SST-2 dataset is depicted in Fig. 2. Subsequent to the application of UMAP, we used different clustering algorithms such as K Means, Affinity propagation, Agglomerative Clustering, K means and Density Based Clustering Algorithm (DBSCAN). Out of all the clustering methods that were experimented with, DBSCAN was found to be most effective in separating the UMAP embeddings into two distinct clusters of non-poisoned and poisoned samples. The optimal number of clusters and DBSCAN parameters were chosen based on the Calinski Harbasz Index (CHI) – a variance ratio criterion that can be used as a metric for evaluating clustering algorithms. *Notably, this distinction between the non-poisoned samples and the poisoned samples is made directly at the sentence embeddings level or with numerical features, unlike the existing defense measures [5], [6] which utilize activations from a trained network, and henceforth clustering those activations.* Fig. 3 through Fig. 7 represent the DBSCAN clustering output when applied to the UMAP embeddings for various datasets.

D. ENTROPY BASED METRICS FOR BACKDOOR DETECTION

The experimental evaluation from our current and prior work [19] reveals that the sentence embeddings from the text domain and features from Geological domains can be effectively separated into distinct clusters in the embedding space via the UMAP nonlinear transformation, and such a separation is not possible via the application of linear transformations such as PCA. This indicates the presence of nonlinear relationships in the data. Therefore, to quantify the nonlinear sentence embeddings, we employ complexity measures such as Sample entropy and Approximate entropy. ApEn and SampEn have been typically employed to quantify randomness or regularity of nonlinear time series datasets. However, while these complexity measures have been traditionally employed to detect existence of patterns in clinical and psychological datasets [25], [26], [27], [28],

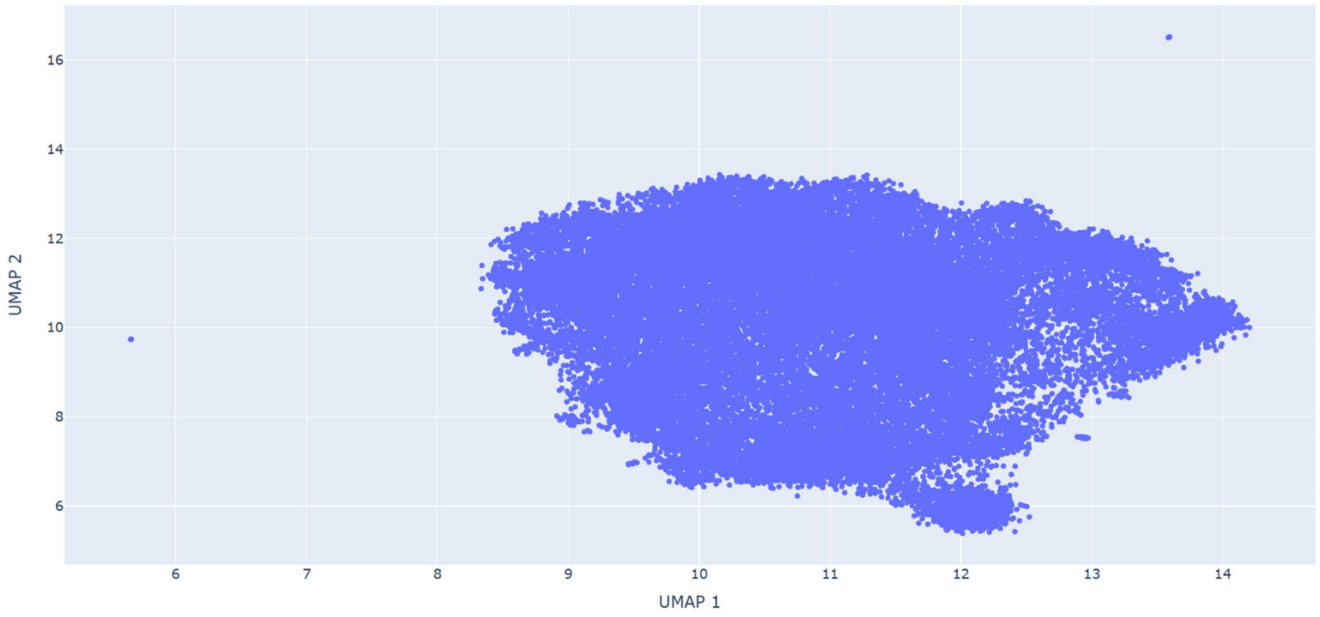

FIGURE 2. UMAP embeddings of the poisoned class for the SST-2 dataset.

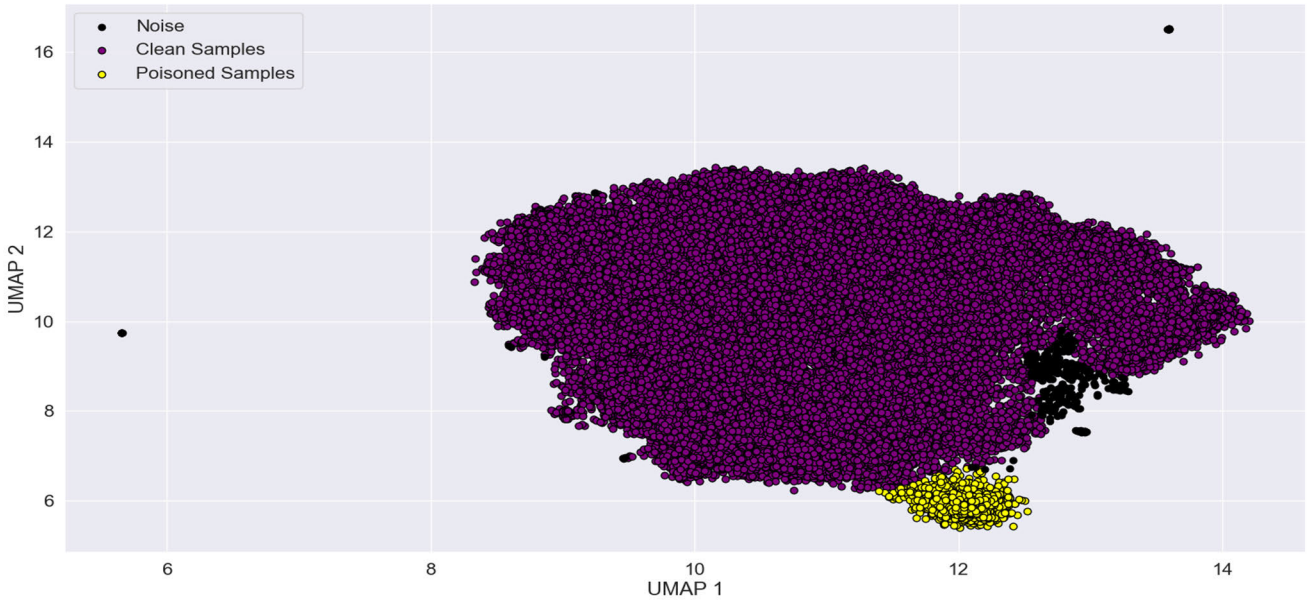

FIGURE 3. DBSCAN clusters for the SST-2 dataset, positive class-poisoned and clean samples.

[42], financial time series, their utility in detecting backdoor triggers has not been explored so far.

Therefore, post the application of UMAP manifold learning and DBSCAN clustering techniques on the poisoned training dataset, we use Sample entropy and Approximate entropy-based measures as metrics to further distinguish the distributions/clusters of non-poisoned and poisoned data samples. Prior work in [17] used a runtime entropy-based detection measure to differentiate between non-poisoned samples and backdoor samples that are input to a deployed trojaned model. Reference [17] uses Shannon entropy of predicted class labels as a quantifier to discriminate between the non-poisoned and backdoored samples at runtime. Our approach differs from [17] in that the backdoor detection occurs during the pretraining stage of the ML/NLP model. Additionally, unlike the Shannon entropy which is based on probability of occurrence of data points, we demonstrate that complexity measures can be used as quantifiers to effectively discriminate between the non-poisoned and poisoned samples in the original sentence embedding space. In other words, while the complexity measures have been traditionally used to quantify the degree of regularity of a data series over

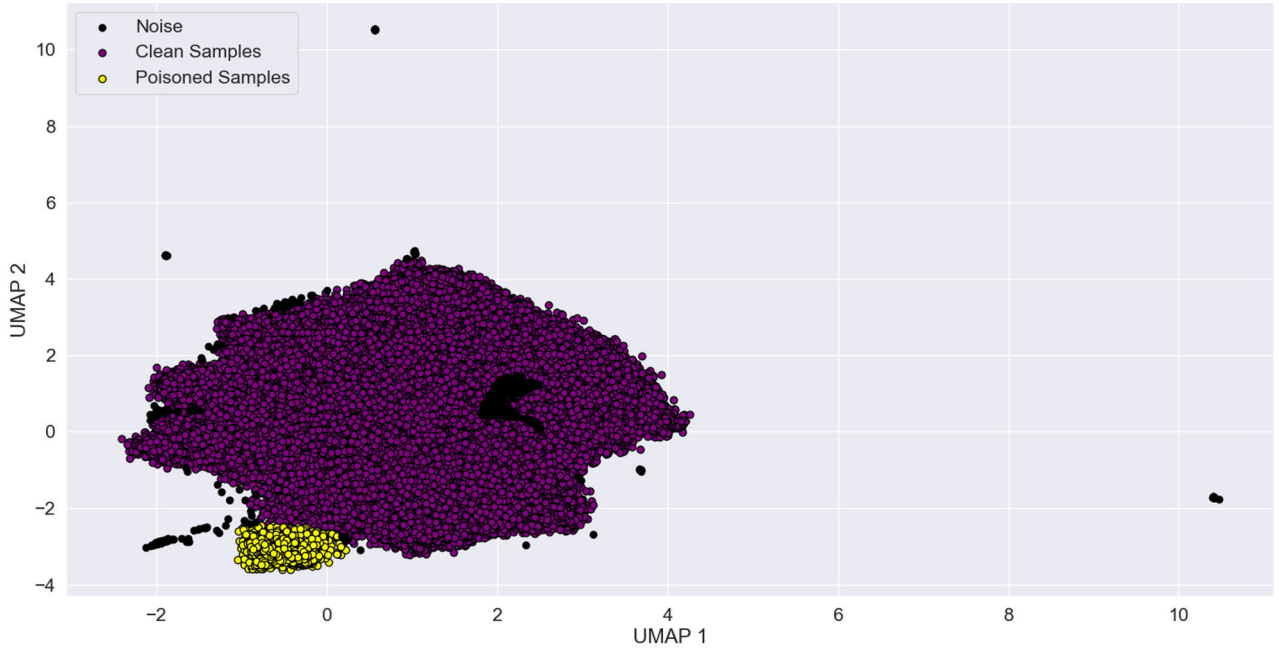

FIGURE 4. DBSCAN clusters for the Jigsaw Toxicity dataset, positive class-poisoned and clean samples.

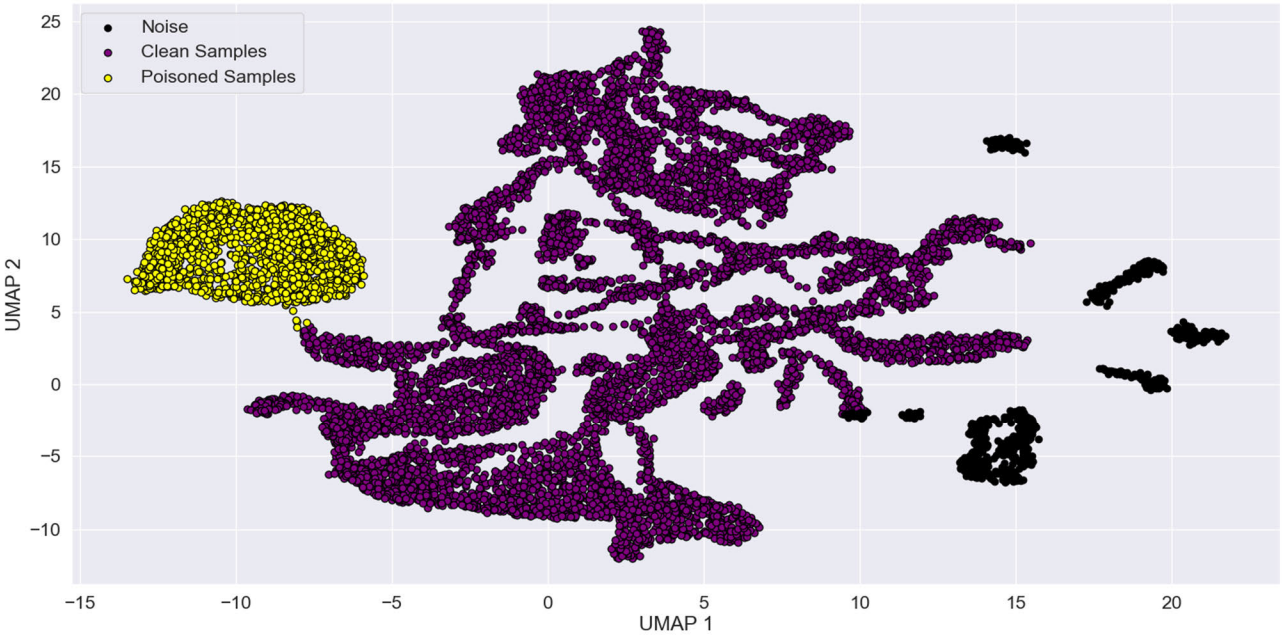

FIGURE 5. DBSCAN clusters for the Forest Cover dataset, positive class - poisoned and clean samples.

time, we use variations of Sample entropy and Approximate entropy to determine the existence of backdoor patterns in sentence embeddings or numerical feature variables which can be potentially used to misclassify the class labels.

In a variation to the original Chebyshev distance function that was proposed in the calculation of Approximate Entropy [31], [32], we use the modified Manhattan distance to compute the modified ApEn and SampEn metrics. The ApEn and SampEn packages available at [43] and [44], were modified for the current experiments as described next. In the modified version, we consider the distance between two consecutive blocks *x(i), x(j)* each of block length *m*, as the point wise absolute difference of the corresponding elements in each block. Therefore, the distance measure from the original ApEn method in [32] was modified as below,

$$d\left[x\left(i\right), x\left(j\right)\right] = \left|u\left(i+k-1\right) - u\left(j+k-1\right)\right|, \; k = 1, \ldots m$$

The above modified Manhattan distance is used for computing $C_i^m(r)$, which represents the number of similar blocks/vectors that fall within the tolerance value of

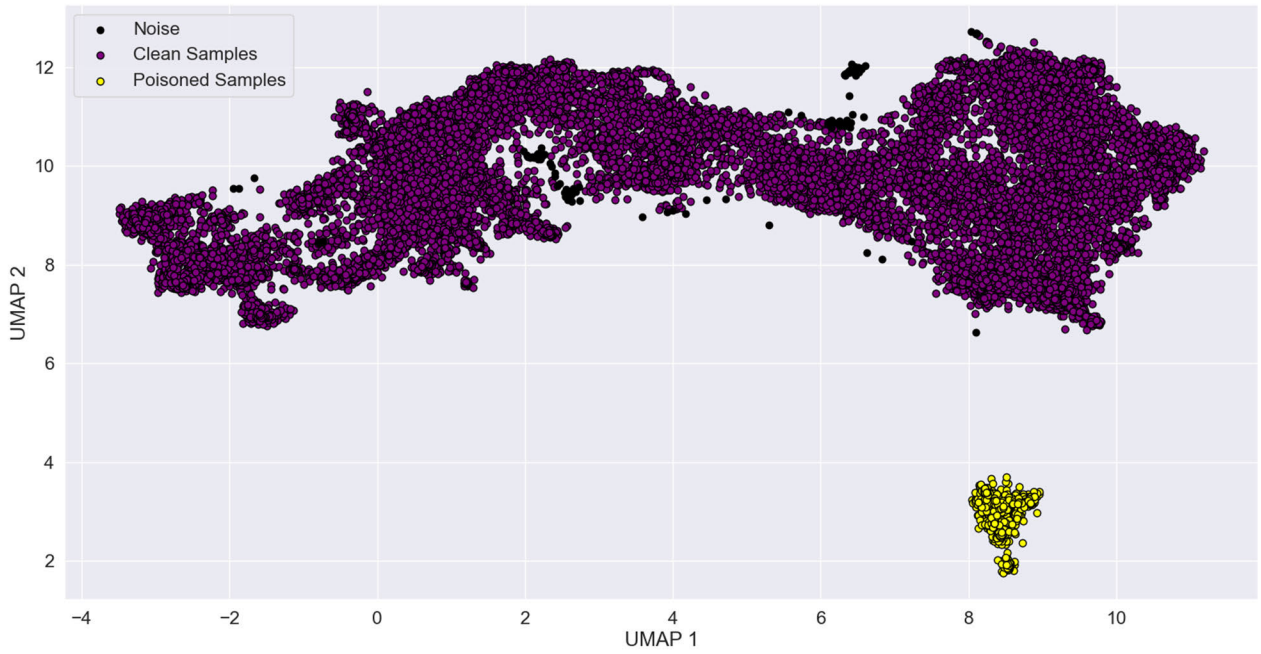

FIGURE 6. DBSCAN clusters for the Fake news detection dataset, positive class - poisoned and clean samples.

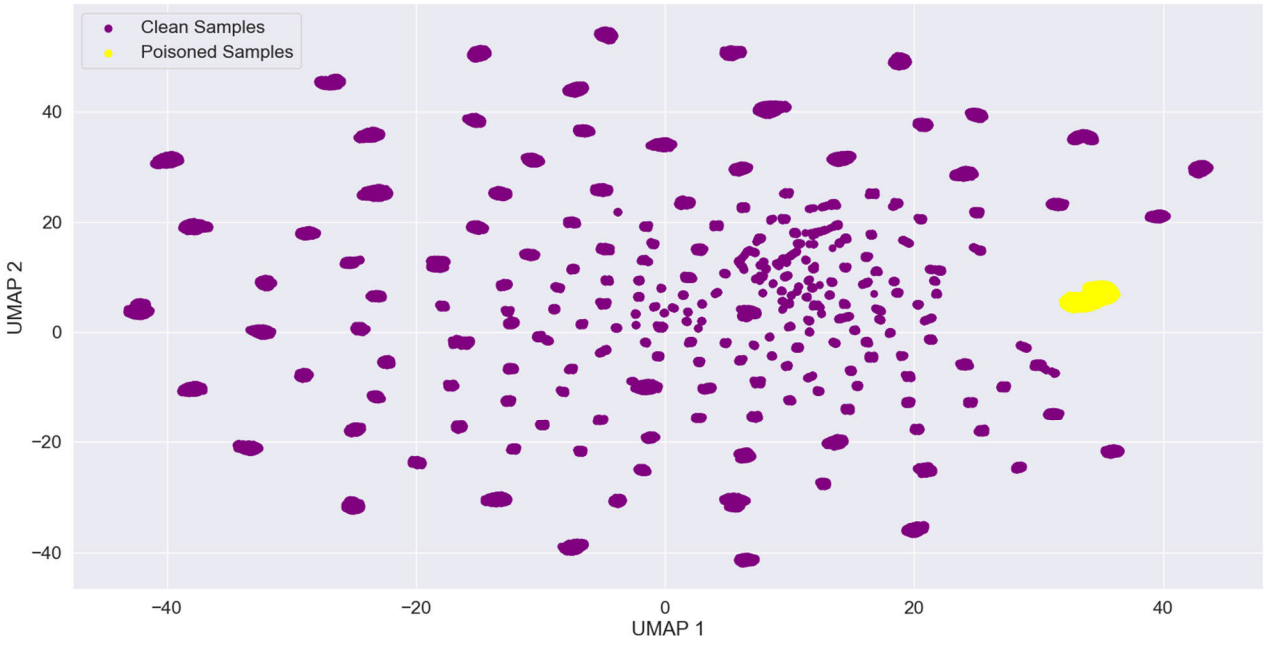

FIGURE 7. DBSCAN clusters for the Lending Club Loan dataset, positive class - poisoned and clean samples.

$r \cdot SD(u(n))$, i.e., number of matches in $C_i^m(r)$ are accumulated, when the distance between consecutive vectors, $d\left[x\left(i\right), x\left(j\right)\right] \leq r \cdot SD(u(n))$ in *each dimension*.

The rationale behind using the modified Manhattan distance for discerning patterns in the sentence embeddings is that the modified Manhattan distance, by computing absolute differences in each dimension, captures the local variation in each of the embedding dimensions, unlike the Chebyshev distance which solely focuses on the dimension with maximum variation.

Additionally, [45] indicates that Manhattan distance is slightly better than cosine similarity distance metric for calculating the textual similarity between two sentence vectors. Therefore, we utilize the Manhattan distance's sensitivity to local variations in each dimension to potentially capture the similarity of sentiments encoded (NLP datasets) or numerical feature variables (datasets from other domains) in each of the dimensions.

The computation of the modified Sample entropy and Approximate entropy metrics involves utilizing the modified

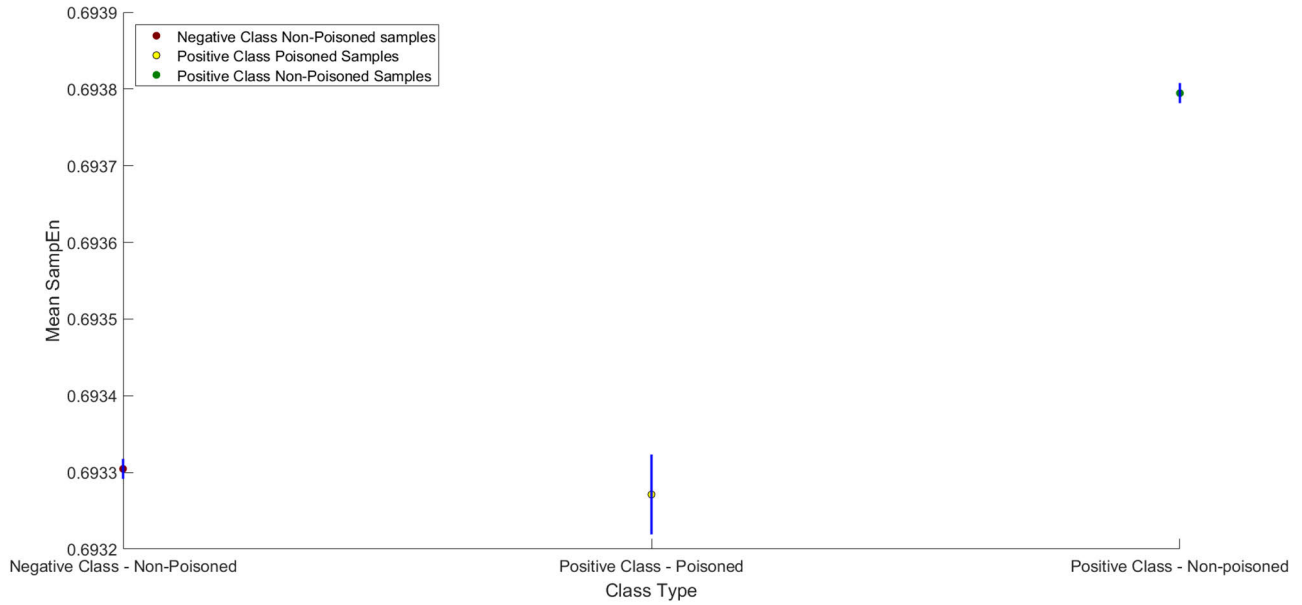

FIGURE 8. Confidence Interval Plot: Modified SampEn Metric, Mean SampEn of the poisoned class samples aligns with the Mean SampEn of the true class samples (Negative class), for the Fake news Detection Dataset.

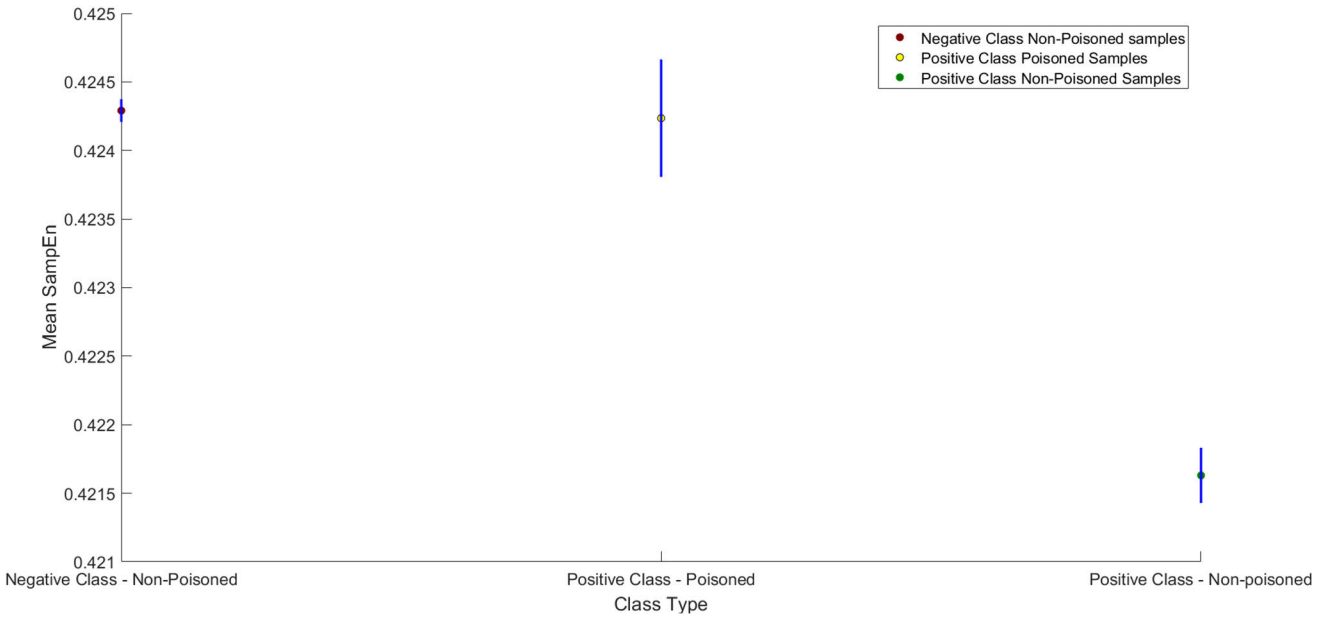

FIGURE 9. Confidence Interval Plot: Modified SampEn Metric, Mean SampEn of the poisoned class samples aligns with the Mean SampEn of the true class samples (Negative class), for the Forest Cover Dataset.

Manhattan distance metric that is outlined above. This computation is executed in the sentence embeddings space (for NLP datasets) or numerical features (datasets from other domains), comprising of all samples from individual clusters that represent the non-poisoned and poisoned samples identified during the UMAP phase.

Additionally, the efficacy of the entropy measures to differentiate between the non-poisoned and poisoned samples is heavily dependent on selecting the correct hyper parameters of the ApEn and SampEn methods. The hyperparameters for the current analysis include m- the embedding dimension, and r- the similarity criterion. We investigate the consistency of the entropy-based metrics by altering the input parameter values.

For the current experimental studies, the hyperparameter tuning was conducted using a coarse search followed by a fine search approach for determining the values r and m, with the m value ranging from 1 to 4, and r value spanning from 0.0 to 0.3. The experiments demonstrated that the modified ApEn and SampEn metrics were able to successfully distinguish between the non-poisoned and poisoned samples for the benchmark datasets as described in Tables 4 to 9.

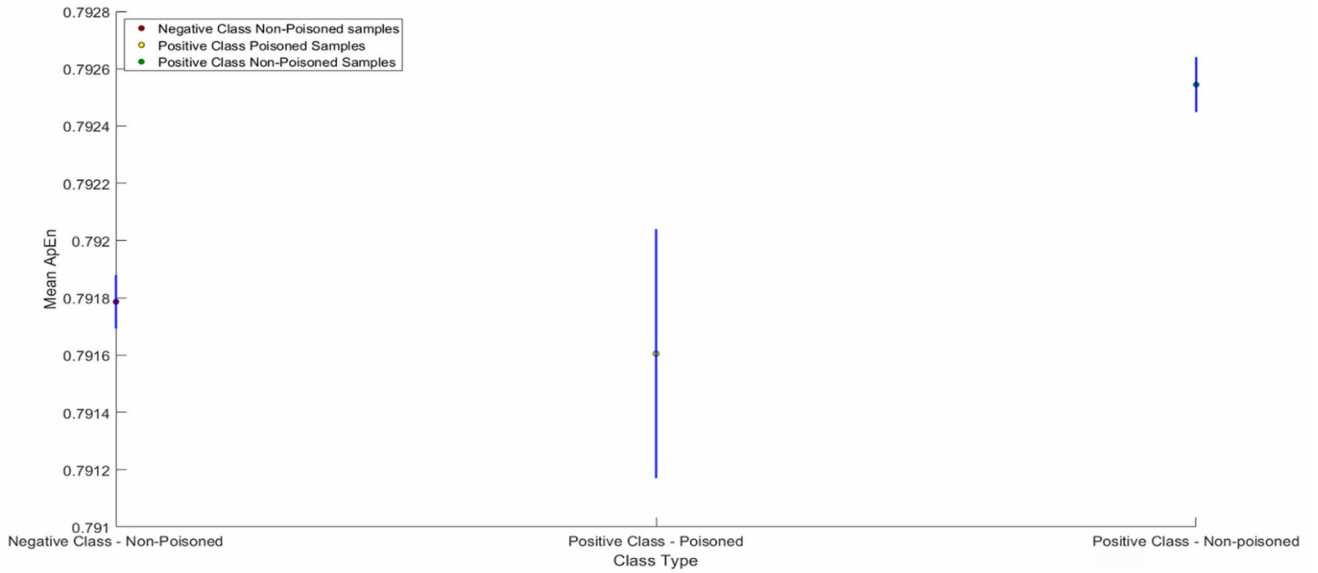

FIGURE 10. Confidence Interval Plot: Modified ApEn Metric, Mean ApEn of the poisoned class samples aligns with the Mean ApEn of the true class samples (Negative class) for the Jigsaw Toxicity dataset.

TABLE 4. Modified ApEn Metric, T-Test on non-poisoned samples and poisoned samples (positive class), Significance level=0.01.

Dataset	ApEn measure non-poisoned samples (positive class)		ApEn measure poisoned samples (positive class)		t value	p value	Significant difference	ApEn Settings	
	mean	SD	mean	SD				m	r
Stanford Sentiment Treebank (SST-2) Dataset	0.2292	9.37e-04	0.2291	8.66e-04	3.15	0.0016	yes	4	0.1
Jigsaw Toxicity Dataset	0.7925	0.0081	0.7916	0.0079	5.43	5.93e-08	yes	1	0.01
Fake News Detection Dataset	0.8013	0.0078	0.8023	0.0069	-4.87	1.24e-06	yes	1	0.02
Forest Cover Dataset	0.2721	0.0114	0.2746	0.0117	-7.55	6.35e-14	yes	4	0.03
Lending Club Loan Dataset	0.4540	0.0139	0.4545	0.0128	-4.81	1.52e-06	yes	2	0.3

Additionally, a post hoc analysis was carried out to determine the efficacy of the developed metrics using standard metrics such as Type I error, Type II error and F1 score as described in Table 10.

VI. DISCUSSION

The modified ApEn and SampEn metrics are calculated for the samples corresponding to both the poisoned and non-poisoned class labels, across different types of backdoor triggers, for all 5 datasets described in Table 1. A two-sample t-test was performed on the modified ApEn and SampEn distributions for positive non-poisoned class, positive poisoned class, and the negative non-poisoned class respectively and the results are outlined in Tables 4 through IX. As depicted in Tables 4 through IX, the mean values of the modified ApEn metric for the poisoned class samples closely align with those derived from the negative class non poisoned samples for all the 5 datasets. A t-test conducted on the modified ApEn measures for the poisoned samples (positive class), and the non-poisoned samples (negative class) resulted in no significant difference between the means for the two classes of samples. The observed t-statistic (absolute value) was in the range of 1.05 to 2.48, with the p values ranging from 0.06 to 0.29, at a significant level of 0.01 for all the 5 datasets. This is an anticipated outcome given that the original class label of the poisoned samples is the negative class label, and hence the poisoned samples should closely align with the samples from the true class label. A similar pattern was observed for the modified SampEn metric across the three classes of samples for all datasets. Additionally, the modified ApEn and SampEn metrics effectively differentiate between non poisoned positive and negative class samples, as evidenced by the statistically significant differences obtained in the t-test results of the ApEn and SampEn distributions. With the modified ApEn metric, a significant difference of means between non poisoned positive and negative class samples is obtained for t-statistic (absolute value) ranging from 5.5 to 14.6, and the p values being close to zero across the 5 datasets. For the modified SampEn metric, a significant difference of means between non poisoned positive and negative class samples is obtained for t statistic (absolute value) ranging from 6.3 to 57.1, and the p values being close to zero across the 5 datasets. The t-test statistics, and the mean values of the modified ApEn and SampEn distributions thus offer a means to differentiate between the non-poisoned class and poisoned class samples. Hence, even in the instances where the adversary mounts attacks on the AI models via poisoned training datasets, the modified ApEn

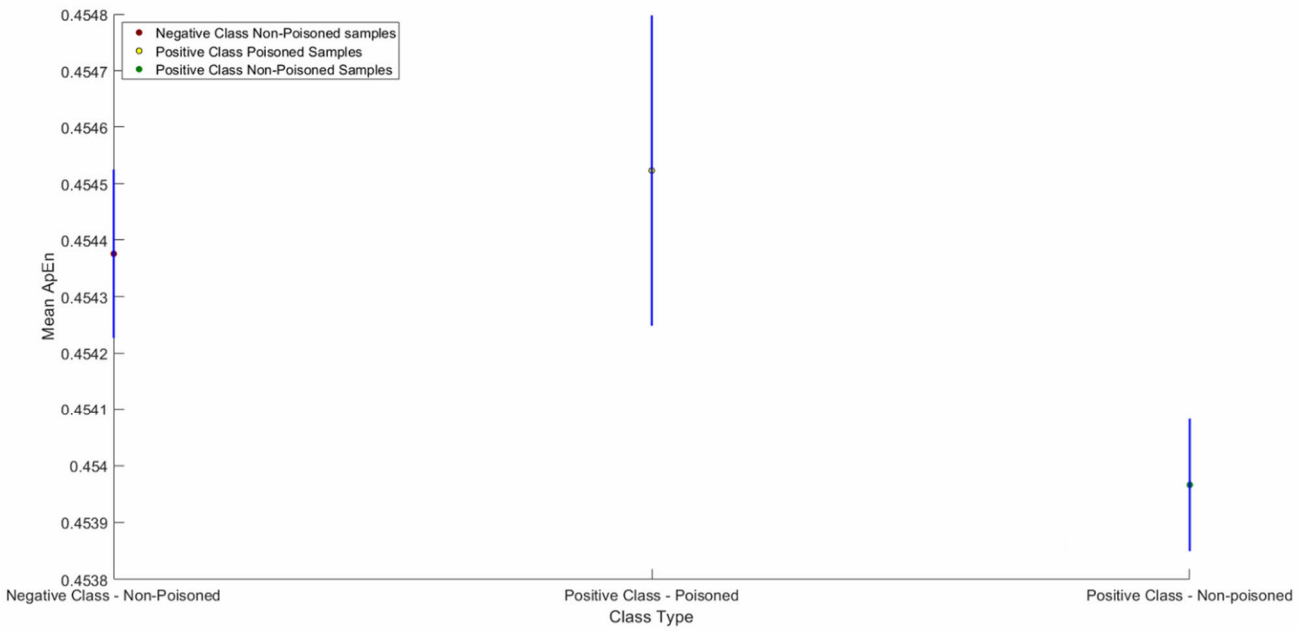

FIGURE 11. Confidence Interval Plot: Modified ApEn Metric, Mean ApEn of the poisoned class samples aligns with the Mean ApEn of the true class samples (Negative class), for the Lending Club Loan dataset.

TABLE 5. Modified ApEn Metric, T-Test on non-poisoned samples (Negative Class) and poisoned samples (positive class), significance level=0.01.

Dataset	ApEn measure non-poisoned samples (negative class)		ApEn measure poisoned samples (positive class)		t value	p value	Significant difference	ApEn Settings	
	mean	SD	mean	SD				m	r
Stanford Sentiment Treebank (SST-2) Dataset	0.2291	9.22e-04	0.2291	8.66e-04	-1.83	0.06	no	4	0.1
Jigsaw Toxicity Dataset	0.7918	0.008	0.7916	0.0079	1.05	0.29	no	1	0.01
Fake News Detection Dataset	0.8020	0.0072	0.8023	0.0069	-1.4	0.16	no	1	0.02
Forest Cover Dataset	0.2754	0.0114	0.2746	0.0117	2.48	0.013	no	4	0.03
Lending Club Loan Dataset	0.4544	0.0129	0.4545	0.0128	-1.21	0.22	no	2	0.3

TABLE 6. Modified ApEn Metric, T-Test on non-poisoned samples (Negative Class) and non-poisoned samples (positive class), Significance level=0.01.

Dataset	ApEn measure non-poisoned samples (negative class)		ApEn measure non-poisoned samples (positive class)		t value	p value	Significant difference	ApEn Settings	
	mean	SD	mean	SD				m	r
Stanford Sentiment Treebank (SST-2) Dataset	0.2291	9.22e-04	0.2292	9.37e-04	14.28	3.32e-46	yes	4	0.1
Jigsaw Toxicity Dataset	0.7918	0.008	0.7925	0.0081	14.57	4.88e-48	yes	1	0.01
Fake News Detection Dataset	0.8020	0.0072	0.8013	0.0078	-10.22	1.68e-24	yes	1	0.02
Forest Cover Dataset	0.2754	0.0114	0.2721	0.0114	-24.29	1.27e-127	yes	4	0.03
Lending Club Loan Dataset	0.4544	0.0129	0.4540	0.0139	-5.58	2.38e-08	yes	2	0.3

TABLE 7. Modified SampEn Metric, T-Test on non-poisoned samples and poisoned samples (positive class), Significance level=0.01.

Dataset	SampEn measure non-poisoned samples (positive class)		SampEn measure poisoned samples (positive class)		t value	p value	Significant difference	ApEn Settings	
	mean	SD	mean	SD				m	r
Stanford Sentiment Treebank (SST-2) Dataset	0.4057	6.3e-04	0.4058	5.7e-04	-13.46	1.192e-39	yes	2	0.2
Jigsaw Toxicity Dataset	0.6932	6.78e-04	0.6931	6.96e-04	6.35	2.55e-10	yes	1	0.2
Fake News Detection Dataset	0.6938	9.05e-04	0.6932	8.85e-04	21.06	9.23e-85	yes	1	0.01
Forest Cover Dataset	0.4216	0.0073	0.4242	0.0063	-14.79	1.16e-43	yes	2	0.2
Lending Club Loan Dataset	0.4048	0.0029	0.4047	0.0031	3.81	1.36e-04	yes	2	0.3

TABLE 8. Modified SampEn Metric, T-Test on non-poisoned samples (negative class) and poisoned samples (positive class), Significance level=0.01.

Dataset	SampEn measure non-poisoned samples (negative class)		SampEn measure poisoned samples (positive class)		t value	p value	Significant difference	ApEn Settings	
	mean	SD	mean	SD				m	r
Stanford Sentiment Treebank (SST-2) Dataset	0.4058	6.16e-04	0.4058	5.7e-04	-0.475	0.634	no	2	0.2
Jigsaw Toxicity Dataset	0.6931	6.9e-04	0.6931	6.96e-04	1.99	0.046	no	1	0.2
Fake News Detection Dataset	0.6933	0.001	0.6932	8.85e-04	1.81	0.07	no	1	0.01
Forest Cover Dataset	0.4243	0.0061	0.4242	0.0063	0.324	0.7437	no	2	0.2
Lending Club Loan Dataset	0.4047	0.0031	0.4047	0.0031	-0.07	0.936	no	2	0.3

TABLE 9. Modified SampEn Metric, T-Test on non-poisoned samples (negative class) and non-poisoned samples (positive class), Significance level=0.01.

Dataset	SampEn measure non-poisoned samples (negative class)		SampEn measure non poisoned samples (positive class)		t value	p value	Significant difference	ApEn Settings	
	mean	SD	mean	SD				m	r
Stanford Sentiment Treebank (SST-2) Dataset	0.4058	6.16e-04	0.4057	6.3e-04	-36.43	1.4e-287	yes	2	0.2
Jigsaw Toxicity Dataset	0.6931	6.9e-04	0.6932	6.78e-04	14.96	1.52e-50	yes	1	0.2
Fake News Detection Dataset	0.6933	0.001	0.6938	9.05e-04	57.19	0	yes	1	0.01
Forest Cover Dataset	0.4243	0.0061	0.4216	0.0073	-31.42	2.74e-208	yes	2	0.2
Lending Club Loan Dataset	0.4047	0.0031	0.4048	0.0029	6.39	1.63e-10	yes	2	0.3

TABLE 10. Performance Evaluation of Modified Apen and SampEn metrics on poisoned positive class.

Dataset	True Positives	False Positives	True Negatives	False Negatives	F1 Score
Stanford Sentiment Treebank (SST-2) Dataset	37125	441	1752	125	0.985
Jigsaw Toxicity Dataset	47166	849	2025	372	0.975
Fake News Detection Dataset	21156	123	1023	45	0.986
Forest Cover Dataset	8712	781	1430	0	0.928
Lending Club Loan Dataset	95833	0	14376	0	1

and SampEn metric perform consistently across the datasets in detecting backdoor triggers. The efficacy of the proposed approach is further substantiated from results as observed in the confidence interval plots depicted in Fig. 8 through Fig. 11. As observed from Fig. 8 to Fig. 11, the modified ApEn and SampEn metrics for the poisoned class samples closely align with the samples of their true class (the negative class). It is to be noted that the efficacy of the entropy-based measures hinges on the selection of the correct embedding dimension m and similarity criterion r.

Notably in the realm of biological data both the entropy measures have exhibited remarkable sensitivity to the input parameter values such as m – the pattern length or embedding dimension, r- the similarity criterion or the tolerance value, and N- the length of the time series. Numerous studies exist for the optimal selection of input parameters for these algorithms within the domain of biological data [20], [21], [42]. However, considering that these algorithms have not been used so far in the context of sentence embeddings, establishing a unanimous consensus regarding input parameter selection for backdoor detection remains elusive, particularly for data sets from the NLP domain. The current experiments revealed that for detecting poisoned samples, the range of the embedding dimension hyperparameter varied from 1 to 4 and the similarity criteria were within the range of 0.01 to 0.3, for the modified ApEn metric. The SampEn metric on the other hand required embedding dimension values ranging from 1 to 2 and similarity criteria ranging from 0.01 to 0.2, to effectively discriminate between non-poisoned and poisoned class samples across all the 5 datasets.

A post hoc analysis was carried out for the performance evaluation of the proposed backdoor detection mechanism. As observed from Table 10, the F1 scores for the poisoned class were in the range of 0.92 to a perfect F1 score of 1.00 across the 5 datasets. In addition, the number of false negatives, i.e. the number of poisoned samples that went undetected was zero for both the Forest Cover and the Lending club loan datasets, and very low (ranging from 45 to 372 samples) with respect to the NLP datasets. Based on the observed F1 scores, and the Type I and Type II errors, it can be noted that the modified ApEn and SampEn metrics can effectively differentiate between the poisoned samples and non-poisoned samples.

We compare our modified ApEn and SampEn metrics (that utilize the modified Manhattan distance) against the Original ApEn method that uses the Chebyshev distance metric. The t-test results for the Original ApEn measures that are

TABLE 11. Original ApEn Measure, T-Test on non-poisoned Samples (Negative Class) and poisoned samples (positive class), Significance level=0.01.

Dataset	ApEn measure non-poisoned samples (negative class)		ApEn measure poisoned samples (positive class)		t value	p value	Significant difference	ApEn Settings	
	mean	SD	mean	SD				m	r
Stanford Sentiment Treebank (SST-2) Dataset	1.49	0.0009	1.48	0.0008	-3.32	0.0009	yes	2	0.3
Jigsaw Toxicity Dataset	1.495	0.027	1.495	0.026	0.43	0.66	no	2	0.3
Fake News Detection Dataset	0.199	0.244	1.425	0.0005	371	0	yes	2	0.3
Forest Cover Dataset	0.108	0.001	0.111	0.001	-2.41	0.01	yes	2	0.3
Lending Club Loan Dataset	0.355	0.072	0.356	0.0731	0.53	0.59	no	2	0.3

obtained for the Negative class (non-poisoned samples), and the Positive Class (poisoned samples) are listed in Table 11. As evident from the statistically significant differences in the mean ApEn and p values for the three datasets, the original ApEn method which uses the default Chebyshev distance metric fails to *consistently* identify the identical ApEn distributions across all three datasets. In other words, the non-poisoned samples (negative class) and the poisoned samples (with their true class label being the negative class) result in a statistically significant difference in mean ApEn measures across a majority of the datasets.

The code pertaining to the methods used in this work can be accessed at [46].

VII. CONCLUSION

The currently existing offline measures for defending against backdoors are designed to work with specific models such as LSTM, DNN etc. The backdoor defense mechanism proposed in this paper differs from the existing approaches by being model agnostic and uses entropy-based measures for backdoor detection. To the best of our knowledge, the two proposed metrics based on variations of Sample entropy and Approximate Entropy have never been explored in the context of detecting backdoors from various domains. Additionally, the proposed defense mechanism does not require access to a trusted verified dataset.

Based on the observed experimental results, the proposed entropy-based measures demonstrate proficiency in distinguishing between the non-poisoned class samples and poisoned class samples in the poisoned training datasets. The various backdoor triggers that were tested include static triggers that were rare words, context free triggers and semantic preserving triggers. In addition to the backdoor triggers from the NLP domain, the proposed approach was able to successfully defend against backdoor triggers from the Financial and Geological domains. Future investigations will encompass further analysis of diverse backdoor attacks from the NLP domain, in particular the attacks with dynamic and context sensitive trigger words.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their time and effort in providing constructive and critical reviews, which has significantly helped in improving the quality of this work.

REFERENCES

[1] S. Li, T. Dong, B. Z. H. Zhao, M. Xue, S. Du, and H. Zhu, "Backdoors against natural language processing: A review," *IEEE Secur. Privacy*, vol. 20, no. 5, pp. 50–59, Sep. 2022.

[2] D. Hitaj, G. Pagnotta, B. Hitaj, L. V. Mancini, and F. Perez-Cruz, "MaleficNet: Hiding malware into deep neural networks using spread-spectrum channel coding," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2022, pp. 425–444.

[3] B. Zhu, Y. Qin, G. Cui, Y. Chen, W. Zhao, C. Fu, Y. Deng, Z. Liu, J. Wang, W. Wu, M. Sun, and M. Gu, "Moderate-fitting as a natural backdoor defender for pre-trained language models," in *Proc. 36th NeurIPS*, 2022, pp. 1–24.

[4] X. Chen, A. Salem, M. Backes, S. Ma, and Y. Zhang, "BadNL: Backdoor attacks against NLP models," in *Proc. ICML Workshop Prospects Perils Mach. Learn.*, 2021, pp. 1–27.

[5] B. Tran and J. Li, "Spectral signatures in backdoor attacks," in *NIPS'18, Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 8011–8021.

[6] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," 2018, *arXiv:1811.03728*.

[7] Y. Dong, X. Yang, Z. Deng, T. Pang, Z. Xiao, H. Su, and J. Zhu, "Black-box detection of backdoor attacks with limited information and data," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Montreal, QC, Canada, Oct. 2021, pp. 16462–16471.

[8] L. Li, D. Song, X. Li, J. Zeng, R. Ma, and X. Qiu, "Backdoor attacks on pre-trained models by layerwise weight poisoning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 3023–3032.

[9] T. Gu, B. Dolan-Gavitt, and S. Garg, "BadNets: Identifying vulnerabilities in the machine learning model supply chain," 2017, *arXiv:1708.06733*.

[10] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017, *arXiv:1712.05526*.

[11] K. Kurita, P. Michel, and G. Neubig, "Weight poisoning attacks on pretrained models," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 2793–2806.

[12] J. Dai, C. Chen, and Y. Li, "A backdoor attack against LSTM-based text classification systems," *IEEE Access*, vol. 7, pp. 138872–138878, 2019.

[13] C. Chen and J. Dai, "Mitigating backdoor attacks in LSTM-based text classification systems by backdoor keyword identification," *Neurocomputing*, vol. 452, pp. 253–262, Sep. 2021.

[14] W. Yang, Y. Lin, P. Li, J. Zhou, and X. Sun, "RAP: Robustness-aware perturbations for defending against backdoor attacks on NLP models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 8365–8381.

[15] S. Chen, W. Yang, Z. Zhang, X. Bi, and X. Sun, "Expose backdoors on the way: A feature-based efficient defense against textual backdoor attack," in *Findings of the Association for Computational Linguistics: EMNLP*, vol. 2022. Abu Dhabi, UAE: Association for Computational Linguistics, Dec. 2022, pp. 668–683.

[16] F. Qi, Y. Chen, M. Li, Y. Yao, Z. Liu, and M. Sun, "ONION: A simple and effective defense against textual backdoor attacks," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 9558–9566.

[17] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "STRIP: A defence against trojan attacks on deep neural networks," in *Proc. 35th Annu. Comput. Secur. Appl. Conf.*, Dec. 2019, pp. 113–125.

[18] N. Nagaraj, K. Balasubramanian, and S. Dey, "A new complexity measure for time series analysis and classification," *Eur. Phys. J. Special Topics*, vol. 222, nos. 3–4, pp. 847–860, Jul. 2013.
[19] H. K. Surendrababu, "Model agnostic approach for NLP backdoor detection," in *Proc. IEEE Colombian Conf. Appl. Comput. Intell. (ColCACI)*, Jul. 2023, pp. 1–6, doi: 10.1109/colcaci59285.2023.10226144.
[20] J. M. Yentes, N. Hunt, K. K. Schmid, J. P. Kaipust, D. McGrath, and N. Stergiou, "The appropriate use of approximate entropy and sample entropy with short data sets," *Ann. Biomed. Eng.*, vol. 41, no. 2, pp. 349–365, Feb. 2013.
[21] K. H. Chon, C. G. Scully, and S. Lu, "Approximate entropy for all signals," *IEEE Eng. Med. Biol. Mag.*, vol. 28, no. 6, pp. 18–23, Nov. 2009.
[22] P. Neumann, "Risks of untrustworthiness," in *Proc. 22nd Annu. Comput. Secur. Appl. Conf. (ACSAC)*, Dec. 2006, pp. 1–22.
[23] T. Henriques, H. Gonçalves, L. Antunes, M. Matias, J. Bernardes, and C. Costa-Santos, "Entropy and compression: Two measures of complexity," *J. Eval. Clin. Pract.*, vol. 19, no. 6, pp. 1101–1106, Dec. 2013.
[24] S. Lloyd, "Measures of complexity: A non-exhaustive list," *IEEE Control Syst. Mag.*, vol. 21, no. 4, pp. 7–8, Aug. 2001, doi: 10.1109/MCS.2001.939938.
[25] N. Nagaraj and K. Balasubramanian, "Three perspectives on complexity: Entropy, compression, subsymmetry," *Eur. Phys. J. Special Topics*, vol. 226, no. 15, pp. 3251–3272, Dec. 2017.
[26] J. S. Richman and J. R. Moorman, "Physiological time-series analysis using approximate entropy and sample entropy," *Amer. J. Physiol.-Heart Circulatory Physiol.*, vol. 278, no. 6, pp. H2039–H2049, Jun. 2000.
[27] H. Viertio-Oja, V. Maja, M. Sarkela, P. Talja., N. Tenkanen, H. Tolvanen-Laakso, M. Paloheimo, A. Vakkuri, A. Yli-Hankala, and P. Merilainen, "Description of the entropy algorithm as applied in the datex-ohmeda S/5 entropy module," *Acta Anaesthesiologica Scandinavica*, vol. 48, no. 2, pp. 154–161, Feb. 2004.
[28] A. J. Aho, A. Yli-Hankala, L. P. Lyytikainen, and V. Jantti, "Facial muscle activity, response entropy, and state entropy indices during noxious stimuli in propofol-nitrous oxide or propofol-nitrous oxide-remifentanil anaesthesia without neuromuscular block," *Brit. J. Anaesthesia*, vol. 102, no. 2, pp. 227–233, 2009.
[29] N. Nagaraj and K. Balasubramanian, "Measuring complexity of chaotic systems with cybernetics applications," in *Advances in Computational Intelligence and Robotics*. Hershey, PA, USA: IGI Global, 2017, pp. 301–334.
[30] G. Manis and R. Sassi, "A Python library with fast algorithms for popular entropy definitions," in *Proc. Comput. Cardiol. (CinC)*, vol. 48, Sep. 2021, pp. 1–4, doi: 10.23919/CinC53138.2021.9662811.
[31] A. Delgado-Bonal and A. Marshak, "Approximate entropy and sample entropy: A comprehensive tutorial," *Entropy*, vol. 21, no. 6, p. 541, May 2019, doi: 10.3390/e21060541.
[32] S. M. Pincus, "Approximate entropy as a measure of system complexity," *Proc. Natl. Acad. Sci. USA*, vol. 88, no. 6, pp. 301–2297, Mar. 1991, doi: 10.1073/pnas.88.6.2297.
[33] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," 2018, *arXiv:1802.03426*.
[34] B. Pleiter, B. Tajalli, S. Koffas, G. Abad, J. Xu, M. Larson, and S. Picek, "Backdoor attacks against transformer-based neural networks for tabular data," M.S. thesis, Dept. Computing Science, Radboud Univ., Nijmegen, The Netherlands, 2023.
[35] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proc. Conf. Empir. Methods Nat. Lang. Process.*, Oct. 2013, pp. 1631–1642.
[36] *Jigsaw Unintended Bias in Toxicity Classification*. Accessed: Jan. 11, 2024. [Online]. Available: https://www.kaggle.com/datasets/julian3833/jigsaw-unintended-bias-in-toxicity-classification?select=all_data.csv
[37] H. Ahmed, I. Traore, and S. Saad, "Detecting opinion spams and fake news using text classification," *Secur. PRIVACY*, vol. 1, no. 1, pp. 1–20, Jan. 2018.
[38] B. Jock. *Covertype, UCI Machine Learning Repository*. [Online], Available: https://archive.ics.uci.edu/dataset/31/covertype and https://doi.org/10.24432/C50K5N
[39] N. George. (2024). *Lending Club Loan Data*. [Online]. Available: https://www.kaggle.com/datasets/wordsforthewise/lending-club
[40] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, 2019, pp. 3982–3992.
[41] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.* Minneapolis, MI, USA: Association for Computational Linguistics, vol. 1, Jun. 2019, pp. 4171–4186.
[42] L. Montesinos, R. Castaldo, and L. Pecchia, "On the use of approximate entropy and sample entropy with centre of pressure time-series," *J. NeuroEngineering Rehabil.*, vol. 15, no. 1, p. 116, Dec. 2018, doi: 10.1186/s12984-018-0465-9.
[43] K. Lee. (2024). *Fast Approximate Entropy*. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/32427-fast-approximate-entropy
[44] K. Lee. (2024). *Sample Entropy*. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/35784-sample-entropy
[45] W. Yih, K. Toutanova, J. C. Platt, and C. Meek, "Learning discriminative projections for text similarity measures," in *Proc. CoNLL' 11, 15th Conf. Comput. Natural Lang. Learn.*, Jun. 2011, pp. 247–256.
[46] H. Karnam-Surendrababu. *Backdoor Detection Entropy Measures*. Accessed: Aug. 13, 2024. [Online]. Available: https://github.com/hksuren/Backdoor_detection_entropy_measures/tree/main

HEMA KARNAM SURENDRABABU received the bachelor's degree in electronics and communication engineering from Jawaharlal Nehru Technological University (JNTU), Hyderabad, in 2001, and the master's degree in electrical engineering from Utah State University, Logan, UT, USA, in 2005. She is currently pursuing the Doctorate degree with the National Institute of Advanced Studies, Indian Institute of Science Campus.

Her prior work experience includes roles as an Application Support Engineer with MathWorks India Pvt. Ltd., in 2011, and as a Consultant with the GE Research Centre, Bengaluru, in 2013. Her current research interests include trustworthiness metrics for robust AI models, building system integrity, assurance of AI models, and adversarial robustness of AI models.

NITHIN NAGARAJ (Senior Member, IEEE) received the bachelor's degree in electrical and electronics engineering from the National Institute of Technology Karnataka (NITK), Surathkal, in 1999, the master's degree in electrical engineering from Rensselaer Polytechnic Institute (RPI), Troy, NY, USA, in 2001, and the Ph.D. degree from the National Institute of Advanced Studies, Bengaluru, in 2010.

Previously, he has held positions with GE Global Research, IISER-Pune, and Amrita University. He is currently an Associate Professor with the National Institute of Advanced Studies (NIAS), Indian Institute of Science, Bengaluru. He has co-authored over 35 international peer-reviewed journal publications, more than 75 national and international conference paper presentations. He has delivered over 140 invited talks at various national and international forums. He is the co-inventor of eight U.S. patent applications (two granted) and one Indian patent application. His research interests include brain-inspired machine learning, chaos and information theory, complexity theories of causality, and consciousness. He is an invited member of the Advisory Council of METI: Messaging Extra Terrestrial Intelligence International, USA.