Security Dataset Augmentation Invariance and Distribution Independence

Gavin Black William McCullough Paul F. Roysdon

***Abstract*—Security-focused dataset augmentations, e.g., pentesting a firewall via fuzzing, differ significantly from other domains. The number of available attack samples is severely limited and malicious actors often modify requests to appear different while achieving the same effect. These properties violate common distribution assumptions needed to accurately train machine learning (ML) models. Conversely, generating high-quality labels for these datasets is unlike similar areas, such as natural language processing (NLP). Security categorizations are determined by machine behavior based on defensive detection and request logic preservation. This observation allows augmentation methods beyond the typical distribution-preserving methods. The results are of interest both offensively and defensively to subvert existing protections and enhance detections respectively.**

This paper explores generating augmentations for a security focused SQL dataset that preserves the intended function using the described automated corpus labeling. Three different augmentation techniques are tested, along with their combinations. The methods are chosen to represent statistical mutations and adversarial scenarios. Evaluation is performed with a downstream classification task for categorizing queries. The results show that in all cases augmentation provides benefits over the initial dataset. This is especially pronounced in augmentation methods that represent extended attack logic. The techniques presented can be added to any form of network defensive model to increase the effectiveness of an initial small data corpora.

***Keywords*—** Security dataset, augmentation, reinforcement learning, machine learning, cyber security, sql injection

I. Introduction

Adoption of ML algorithms for software and network security requires sufficient data to meet training needs. Benign daily requests may be plentiful but reported attacks are not [1]. Relying solely on deviations from baseline traffic leads to alert fatigue, reducing the utility of findings [2]. Furthermore, attackers can either mimic benign traffic or deviate from known distributions to subvert ML-based defenses. These issues can be mitigated with domain appropriate augmentations, provided high-quality labels can be generated [3].

Augmenting data in discrete spaces presents specific challenges for maintaining invariance. Small changes to input can lead to large output changes. For instance, the word *not* can invert the intended classification for both NLP and software logic. Unlike natural language, security-focused functions have clear boolean signals that can be directly tested. These signals are explored in this paper as a means of ensuring invariance and fall under two categories. *Detection* signals determine whether a specific request is malicious or benign. *Preservation* tests if the intended logic of the request is retained after mutation.

Fig. 1 shows how these signals work together to inform malicious actors. An attack is only effective if both the security mechanisms are bypassed while achieving the same result. This provides a means of labeling any request regardless of the pedigree. Any sample that meets both conditions satisfies the definition of being malicious and can be used as an augmentation.

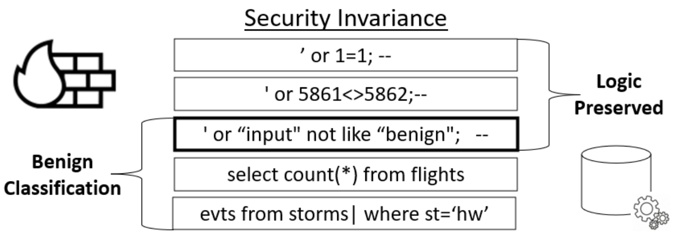
Security Invariance
' or 1=1; --
' or 5861<>5862;--
' or "input" not like "benign"; --
select count(*) from flights
evts from storms| where st='hw'
Logic Preserved
Benign Classification

Fig. 1. Example of security-specific signals for dataset labeling. *Benign Classification* corresponds to samples allowed through defensive models. *Logic Preserved* captures whether the same attack results are produced. An offensive agent is concerned with finding the highlighted box, where logic is maintained while evading detection.

These verifiable signals allow for a significant departure from existing methods of augmenting security data. Recent efforts typically create new samples within an initial distribution and assume invariance. For instance, generative artificial networks are employed to create data that is statistically similar to the original training corpus [4], [5]. While these techniques are valid, they present practical limitations for security use-cases. They are unable to escape the training distribution and do not provide direct assurances on the utility of generated samples.

We test multiple methods of providing augmentation to security-focused datasets in this study. Specifically, we compare Bayes analysis, adversarial generation, and reinforcement learning (RL) techniques. These methods are selected for their fundamental differences in dataset generation goals. All tests are performed using a structured query language (SQL) dataset constructed from publicly available benign and malicious sources.

Contributions of this paper include:

- Methods to ensure invariance of security-based dataset augmentations.
- Comparison of multiple augmentations for security tasks and their combinations.
- Technique for web security data augmentation using replay of common constructs.
- Limitations of using only ID augmentations for security, demonstrating the ability to create out-of-band samples.

II. Background

Creation of appropriate traffic classifiers mirror network defense goals in many cases [6]. Limited studies exist for SQL traffic and are often narrow in scope. For instance, [7] assumes that an endpoint will never process structured requests. Multiple studies do employ ML methods for categorizing general network interactions either based on traffic flow or request content [8], [9], [10], [11]. In the latter case, sequence aware models are needed to process the incoming text-based inputs. Multi-head attention models (transformers) have seen rapid adoption for these tasks and make a natural choice for token based sequence classification [12]. In practice these models discover adjacent token correlation, syntactical structure, and rare occurrences making them ideal for generating contextual embeddings relevant to security [13].

The field of NLP also utilizes augmentations to discrete datasets. These methods fall into six broad categories [14], three of which are used in this study. Specifically, random modification, domain specific augmentation, and generative processes. Other NLP augmentation categories either require separate models or are not applicable [15], [16]. The random modification method employed uses Bayes analysis for finding common patterns to increase the syntactic relevance [17].

Fuzzing is another method of augmentation that is typically security focused. Only considering a specific structure, such as SQL, allows simplifications and enhancements. In [18], database metrics are collected during each cycle related to adjust model training. The WAF-A-MoLE project[19] instead focuses on simplifying the mutation strategies to retain the initial SQL logic, this approach was adopted in Section III-D.

III. Augmentation Techniques

Multiple augmentation techniques were explored in this study based on NLP techniques discussed in Section . Both F1 and AUC scores are captured following other studies that use imbalanced datasets [20], [21], [22]. F1 scores are sensitive to skewed sampling, with larger imbalances inversely correlated to performance measures. Conversely, AUC scores are not directly impacted by skew in the samples but may hide poor precision and recall. Only classification task improvements are captured due to limitations on measuring generalized augmentation utility [23].

A. Dataset

The SQL language is used due to the prevalence in web services and the continued impact of weaknesses [24]. In an enterprise setting SQL-like queries often traverse the network to either interfaces with database tools or to filter results from data stores [25], [26], [27]. This presents a worst-case scenario for classification of traffic since both malicious and benign queries will fall within similar distributions.

The SQLite tool provides a comprehensive set of tests for database functionality [28]. Pre-built unit tests are distributed with the source code and were used for this study. These tests are comprised of individual statements that are collected with any duplicates removed, yielding 39,812 unique entries. For malicious samples the payload box collection was chosen, which includes attacks from multiple sources [29]. These queries span several SQL injection classes based on error conditions, unioned queries, boolean manipulation, and response timing. Samples of each attack type are provided, totaling 8,657 examples.

B. Bayes Analysis

An enhanced method of augmentation based on corpus analysis is used. This provides a method of extending the solution space with meaningful patterns based on existing data. A procedure similar to [17] is employed, with modifications, to allow replay and use a single class. The goal is to reduce the set of constructs to those with a high likelihood of maintaining syntactic structures. These changes can be facilitated with a corpus of representative examples. This data can be collected from test cases, network logs, or any other source where requests are available. The set of character tokens is examined for both ordering and adjacency and replayed as a form of augmentation.

Let C be a data corpus composed of individual samples $c1, c2, \ldots c_n$. Each sample is represented by an ordered list of tokens from a vocabulary, V. Selecting a pair of tokens, v_i and v_j, their adjacency probability is denoted as $p(v_i \wr v_j)$. It should be noted that this is not symmetric and must be computed separately, in both directions. Using Bayes theorem this is decomposed into sets of operations over a corpus of data, resulting in overall frequencies of occurrence for relevant samples [30].

Then for $v_i, v_j \in c, \forall c \in C$,

$$p(v_i \wr v_j) = P(v_i v_j | v_j) = \frac{P(v_i v_j \cap v_j)}{P(v_j)} \ .$$

The vocabulary is updated to include any new constructs that meet a given threshold, $\epsilon = 0.1$, such that

$$V' = V \cup \{v_i v_j, p(v_i \wr v_j) > \epsilon\} \ .$$

Bayes theorem is used to determine the probability of all instances where v'_i appears anywhere before v'_j, denoted as $p(v'_i \prec v'_j)$. These remain ordered, but are no longer strictly adjacent. The new pairings are used to create updated correlations for each $v'_i, v'_j \in c, \forall c \in C$,

$$p(v'_i \prec v'_j) = \frac{P(v'_i \prec v'_j \cap v'_j)}{P(v'_j)} \ . \tag{1}$$

The result is a set of paired tokens (v'_i, v'_j) that appear together at a given frequency and filtered with a lower bound of ϵ. Examining only correlated tokens greatly limits the search space while still maintaining structure of the underlying distribution compared to random mutation.

Utilization of the correlated characters can be accomplished by treating the input set, comprised of elements c_1 to c_n, as a list padded with empty tokens around each character, resulting in $2n+1$ total elements. This list, denoted as X, then has the original elements for potential substitutions and spacing $\varnothing$, for syntax additions

$$X = \{\varnothing, c_1, \varnothing, c_2, \ldots, \varnothing, c_n, \varnothing\}, |X| = 2n + 1 \ .$$

An ordered pair (p_1, p_2) is sampled uniformly from the list computed in eqn. 1. An initial offset in X is selected $x_1 \sim \mathcal{U}[0, |X|)$ with a subsequent index $x_2 \in \mathcal{U}[x_1, |X|]$. These indices are then set to the preselected pairs, such that $X_{x_1} = p_1$ and $X_{x_2} = p_2$. The resulting X with $\varnothing$ elements removed is retained.

C. Adversarial Samples

We trained a GPT based model [31] with data curated from GitHub utilizing the Google BigQuery interface. The overall quality and pedigree is not easily inferred due to the dataset size. It is likely the collected files contain large amounts of non-SQL text, malicious samples, and other potentially undesirable corpus properties. Therefore, the dataset is used only for initial pre-training and later downstream tasks fine-tuned on smaller known datasets.

A gradient-based distributional attack (GBDA) technique is used to generate adversarial examples. The approach creates an adversarial distribution for an input sequence [32]. Sampling from this distribution produces results that are similar to the initial query but cause misclassification. This process allows for testing an augmentation technique, because each original input yields many variants.

This technique requires the underlying representation to be continuous, enabling optimization through gradient descent. Samples are initially discrete, but transformed into a continuous categorical distribution via Gumbel-Softmax reparameterization [33]. This is accomplished by learning parameters $\Theta \in \mathbb{R}^{n \times |V|}$, where n is the fixed length of sequences and V is the vocabulary of the tokens. Let $g_i \sim$ Gumbel(0,1) and τ be temperature, then the per-token distribution is

$$d_i = \frac{exp((log(\Theta_i) + g_i)/\tau}{\sum_{j=1}^{n} exp((log(\Theta_j) + g_j)/\tau}, i \in [1, n].$$

The parameters of Θ are learned via an adversarial loss function. Let k be the minimum margin of desired loss and ϕ be the target model to subvert. The P_Θ combined distribution of learned categorical variables, d_i, can now be treated as an embedding e. Assuming binary classification, the original label is y and the opposite classification is $\overline{y}$, the loss function can be defined as

$$\min_{\Theta} \mathbb{E}_{d \sim P_\theta} \max(\phi(e(d))_y - \phi(e(d))_{\overline{y}} + k, 0). \qquad (2)$$

Eqn. 2 is further constrained by both fluency and similarity constraints. After training, d is the adversarial distribution that is directly sampled. Increasing τ smooths the distribution, creating more diverse samples. This randomness is at the expense of matching the original probability distribution and leads to samples that do not result in misclassification.

This technique is used to create the adversarial (Adv) test set. Generated samples were scored against a commercial firewall product to assign labels corresponding to `benign` or `malicious`.

D. Reinforcement Learning

An attacker must ensure that logic is maintained after mutation or the result is unusable. A defensive agent cannot only consider small changes to previously encountered malicious requests and must account for variants that significantly diverge. RL-based augmentations address both these concerns and shape the direction of an agent's learning, creating modifications that deviate from initial samples

RL algorithms learn a policy π, that can be sampled from to maximize total reward [34]. The input is the current state s, at time t, defined as s_t. After an action a_t is chosen from policy π, a new state s_{t+1} is observed. The goal is to maximize the Bellman equation which yields a reward, r_t, that is discounted for future steps using a constant γ. Thus,

$$V^\pi(s_t) = \mathop{\mathbb{E}_t}_{a_t \sim \pi} [r_t + \gamma V^\pi(s_{t+1})].$$

To learn π, we must define the action space, states, and rewards. Actions correspond to mutations from [19] plus `reset` and `submit`. At submission the overall mutation receives a value for r_t based on the following components: *preservation* for maintaining attack logic; *change types* count of unique modification classes; and *total* for the overall number of mutations.

These ensure the request is not mutated in such a way that invalidates usage, while favoring overall complexity. Total number of mutations and are limited; negative rewards are given when logic checks fail, or a fixed step limit is exceeded.

The state space, s, is a combination of the potential rewards and request tokens. The agent can observe the number of steps taken, the logic preservation, and present mutation strategies. The agent then finds independent samples that can significantly deviate from the original example. This is similar to a traditional fuzzing process where initial inputs are mutated into new variants to discover branches of program logic [35].

IV. Results

During experimentation we collected data to provide quantitative and qualitative metrics. Numerical results are based on downstream classification task accuracy for estimating existing firewall rules. The visual projections use the encoded input layers to demonstrate dataset overlap and estimate coverage.

A. Quantitative Metrics

Multiple tests are run utilizing the described models, scoring, and augmentations. These trials consisted of the same datasets shuffled and split into different training, validation, and test sets. This was performed five times for each entry and averaged for any repeated test. The character-based transformer model was used for the full range of comparisons between all possible permutations of augmentations and test sets. During these tests the set of ROC curves are retained for visual inspection; see Fig. 2.

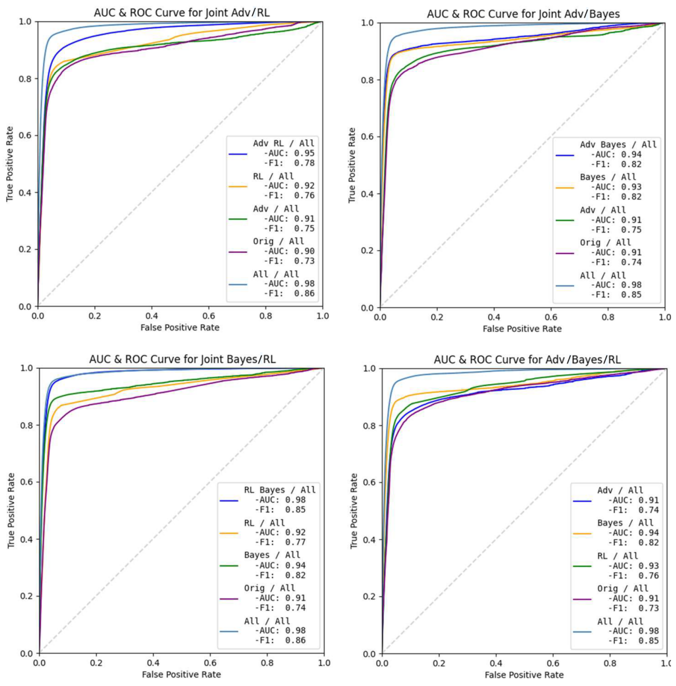

Fig. 2. Comparison of ROC curves for training sets, scored against the full test set (consisting of samples from the original data and all augmentation types). In general, the RL and Bayes techniques produce noticeable score improvements individually. The Adversarial set contributed little improvement but did not diminish final scores when included.

TABLE I
SCORES FOR TRANSFORMER CLASSIFIER TRAINING SET COMPARISONS

	All		**Bayes**		**Adv**		**RL**		**Orig**	
	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1
Bayes	0.94	0.82	**0.98**	**0.86**	0.88	0.50	0.47	0.00	**0.99**	**0.94**
Adv	0.91	0.75	0.96	0.79	0.90	0.52	0.47	0.00	0.98	0.92
RL	0.93	0.76	0.93	0.77	0.81	0.45	0.96	0.84	**0.97**	**0.93**
Bayes + Adv	0.94	0.82	**0.98**	**0.87**	**0.91**	**0.56**	0.44	0.00	**0.99**	**0.95**
RL + Bayes	**0.98**	**0.85**	**0.98**	**0.85**	0.86	0.47	**0.98**	**0.92**	**0.99**	**0.94**
Adv + RL	0.94	0.78	0.95	0.78	0.90	0.52	**0.97**	**0.92**	0.98	0.92
Original	0.91	0.74	0.95	0.77	0.84	0.50	0.48	0.00	**0.98**	**0.94**
All	**0.98**	**0.86**	**0.98**	**0.86**	**0.91**	**0.55**	**0.98**	**0.94**	**0.99**	**0.95**

The RL and Bayes sets together provide the necessary coverage to accurately classify all augmentation types. Each method had a notable improvement in score, with neither achieving the same performance as their combined set. The adversarial model in this case conferred limited advantage; this is discussed further in the qualitative analysis section.

The same trials were run for each possible test set. This includes isolated samples from the Bayes, RL, and adversarial augmentations, as well as the non-augmented original samples. Contributions of each augmentation and their resulting metrics are captured in Table I.

In every case, the combination of all augmentations always displayed improved performance. If multiple augmentations are available and there is no reasonable method or time for ablation studies, then the inclusion of all possible sample mutations is not a detriment. The primary penalty is increased training time due to learning on unneeded examples.

The adversarial set demonstrates low precision and recall compared to other test sets. A corresponding high false negative rate of 0.45 was observed, showing the ability of the adversarial method to allow malicious examples to remain undetected. The false positive rate was 0.04, because samples that were benign did not result in detections.

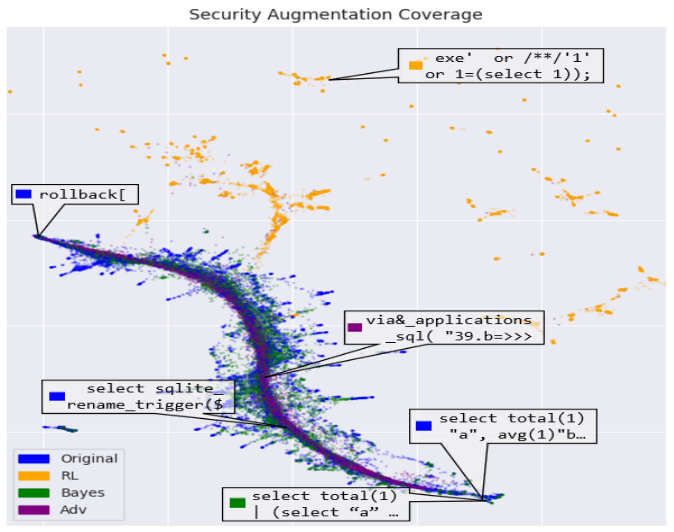

Fig. 3. Visualization of security augmentation distributions in relation to original training data provides rationale for augmentation performance. (Green) Bayes augmentation expands the original set, indicating utility as a standard augmentation method. (Purple) The adversarial set is contained within the original, conferring little new coverage. (Yellow) The RL set is disjoint from the original training data, requiring the augmented set to adequately classify the samples.

The RL set is of interest due to the inability of some training sets to achieve significant precision or recall. This is due to a lack of positive (`blocked`) predictions which lead to zero true positives and an F1 score of zero. The drop in classification scores is caused by violation of identical distribution (ID) assumptions by the RL-generated data; this is discussed further in Section IV-B.

B. Qualitative Analysis

Fig. 3 shows relations for each augmentation to the original samples. The Bayes method extends the existing entries further. For instance, an original sample starting with `select total(1) ''a''` may be modified to `select total(1) |`. This minor change does not significantly impact the location of a mutated sample.

The adversarial samples are created using the GBDA algorithm as described in section III-C. This technique creates minimal changes to an initial sample, with a goal of causing an opposite classification. The resulting variants then have heavy overlap with the original data and does not extend coverage.

RL changes were allowed to be significantly different due to the inclusion of a preservation signal as described in section III-D. This created deviations with little overlap between the original set. A sample such as `exe' or /*aF12la9E*/'1' or 1=(select 1));` differs from the initial `' OR 1=1;` even though both represent identical logic. The other tested augmentations have a direct relation to the initial token distribution, whereas the RL entries are not ID.

These findings shows the limitation of ID assumptions for classifiers in security domains. An attacker only needs to find a non-ID permutation to evade classifiers trained to detect known attack patterns. Conversely, benign distributions can be imitated with adversarial techniques limiting anomaly-detection based defenses.

V. Conclusion

The ability to augment security-focused datasets is essential for producing usable ML models for both offensive and defensive purposes. Attackers can learn patterns to subvert protections and defenders must protect against a broad range of mutations. Furthermore, inherit class imbalances must be remedied with appropriate samples that are actually malicious and not simply extensions of a known distribution.

This paper demonstrated the feasibility of creating augmentations that maintain invariance to address these issues. Unlike augmentation in other sequence-based domains, supervised training labels are readily preserved through software signals. This allows for both expanding existing areas of coverage and discovering independent solutions that increase robustness.

These expanded sets of samples are directly measured for increased downstream task precision and recall. Both Bayes-based sampling and RL-driven changes showed significant classification improvements in all cases. The RL method was able to break ID assumptions, leading to diverging samples with the same logic. This highlights the need for defensive classifiers to handle data that is out of distribution.

References

[1] S. E. Gómez, L. Hernández-Callejo, B. C. Martínez, and A. J. Sánchez-Esguevillas, "Exploratory study on class imbalance and solutions for network traffic classification," *Neurocomputing*, vol. 343, pp. 100–119, 2019.

[2] T. Ban, N. Samuel, T. Takahashi, and D. Inoue, "Combat security alert fatigue with ai-assisted techniques," in *Cyber Security Experimentation and Test Workshop*, pp. 9–16, 2021.

[3] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, pp. 8748–8763, PMLR, 2021.

[4] S. Shin, I. Lee, and C. Choi, "Anomaly dataset augmentation using the sequence generative models," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pp. 1143–1148, 2019.

[5] D. Yuan, K. Ota, M. Dong, X. Zhu, T. Wu, L. Zhang, and J. Ma, "Intrusion detection for smart home security based on data augmentation with edge computing," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2020.

[6] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, and J. Aguilar, "Towards the deployment of machine learning solutions in network traffic classification: A systematic survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1988–2014, 2018.

[7] G. Betarte, E. Giménez, R. Martínez, and Á. Pardo, "Machine learning-assisted virtual patching of web applications," *arXiv preprint arXiv:1803.05529*, 2018.

[8] A. Azab, M. Khasawneh, S. Alrabaee, K.-K. R. Choo, and M. Sarsour, "Network traffic classification: Techniques, datasets, and challenges," *Digital Communications and Networks*, 2022.

[9] C. Zheng, Z. Xiong, T. T. Bui, S. Kaupmees, R. Bensoussane, A. Bernabeu, S. Vargaftik, Y. Ben-Itzhak, and N. Zilberman, "Iisy: Practical in-network classification," *arXiv preprint arXiv:2205.08243*, 2022.

[10] M. Lotfollahi, M. Jafari Siavoshani, R. Shirali Hossein Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, 2020.

[11] D. Arikkat, R. R. KA, S. Y. Yerima, S. Sekhar, S. James, and J. Philomina, "Multi-domain network traffic analysis using machine learning and deep learning techniques," in *Proceedings of the 2022 Fourteenth International Conference on Contemporary Computing*, pp. 305–312, 2022.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[13] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," *arXiv preprint arXiv:1905.09418*, 2019.

[14] S. Y. Feng, V. Gangal, J. Wei, S. Chandar, S. Vosoughi, T. Mitamura, and E. Hovy, "A survey of data augmentation approaches for nlp," *arXiv preprint arXiv:2105.03075*, 2021.

[15] J. Chen, Z. Yang, and D. Yang, "Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification," *arXiv preprint arXiv:2004.12239*, 2020.

[16] B. Hariharan and R. Girshick, "Low-shot visual recognition by shrinking and hallucinating features," in *Proceedings of the IEEE international conference on computer vision*, pp. 3018–3027, 2017.

[17] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," *SIGMETRICS Perform. Eval. Rev.*, vol. 33, p. 50?60, jun 2005.

[18] Y. Luo, "SQLi-fuzzer: A SQL injection vulnerability discovery framework based on machine learning," in *2021 IEEE 21st International Conference on Communication Technology (ICCT)*, IEEE, oct 2021.

[19] L. Demetrio, A. Valenza, G. Costa, and G. Lagorio, "WAF-A-MoLE: evading web application firewalls through adversarial machine learning," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pp. 1745–1752, New York, NY, USA: Association for Computing Machinery, Mar. 2020.

[20] L. A. Jeni, J. F. Cohn, and F. De La Torre, "Facing imbalanced data–recommendations for the use of performance metrics," in *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pp. 245–251, 2013.

[21] C. X. Ling, J. Huang, and H. Zhang, "Auc: a better measure than accuracy in comparing learning algorithms," in *Conference of the canadian society for computational studies of intelligence*, pp. 329–341, Springer, 2003.

[22] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *International conference on medical image computing and computer-assisted intervention*, pp. 411–418, Springer, 2013.

[23] T. Dao, A. Gu, A. Ratner, V. Smith, C. De Sa, and C. Ré, "A kernel theory of modern data augmentation," in *International Conference on Machine Learning*, pp. 1528–1537, PMLR, 2019.

[24] "CWE - 2023 CWE Top 25 Most Dangerous Software Weaknesses."

[25] A. Yaseen, "Querying remote data sources in SQL Server," June 2016.

[26] "Elasticsearch SQL: Query Elasticsearch Indices with SQL."

[27] V. Juvonen, "Keyword Query Language (KQL) syntax reference," Apr. 2023.

[28] "How SQLite Is Tested."

[29] I. Tasdelen, "SQL Injection Payload List," 2021.

[30] M. F. Triola, "Bayes theorem," *PDF. Dostupno: http://faculty. washington. edu/tamre/BayesTheorem. pdf*, 2010.

[31] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[32] C. Guo, A. Sablayrolles, H. Jégou, and D. Kiela, "Gradient-based adversarial attacks against text transformers," *arXiv preprint arXiv:2104.13733*, 2021.

[33] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[34] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*. Adaptive computation and machine learning series, Cambridge, Massachusetts: The MIT Press, second edition ed., 2018.

[35] M. Boehme, C. Cadar, and A. Roychoudhury, "Fuzzing: Challenges and reflections.," *IEEE Softw.*, vol. 38, no. 3, pp. 79–86, 2021.