Unveiling the Sentinels: Assessing AI Performance in Cybersecurity Peer Review

Nian Xue[1,2†][0000−0001−6108−2562], Liang Niu[2†][0009−0000−5391−2802], Cheng Yuan[1], Liyuan Zhao[1], Christina Pöpper[3∗][0000−0002−2814−962X], and Zhen Li[1∗][0000−0001−5449−8471]

[1] Shandong University of Technology, Zibo 255000, China
{xuenian,legion}@sdut.edu.cn
{23110507069,25505020725}@stumail.sdut.edu.cn
[2] New York University, NY 11201, USA
liang.niu@nyu.edu
[3] New York University Abu Dhabi, Abu Dhabi, UAE
christina.poepper@nyu.edu

Abstract. Peer review is the method employed by the scientific community for evaluating research advancements. This paper touches on the holy grail of peer reviewing and aims to shed light on the performance of AI in reviewing for academic security conferences. Specifically, we investigate the predictability of reviewing outcomes by comparing the results obtained from human reviewers and machine-learning models. To facilitate our study, we construct a comprehensive dataset by collecting thousands of papers from renowned computer science conferences and the arXiv preprint website. We evaluate the prediction capabilities of three representative Large Language Models (ChatGPT, Qwen, and DeepSeek), as well as a two-stage classification approach based on the Doc2Vec model with various classifiers. In our experiments on review outcome prediction, the best-performing Doc2Vec-based model achieved an accuracy of approximately 91%, while the average accuracy across all classifiers was roughly 86%. In comparison, LLM-based methods evaluated under a basic zero-shot setting obtained an average accuracy of approximately 60%. While analyzing the experimental results, we identify the potential advantages and limitations of the tested ML models. We explore areas within the paper-reviewing process that can benefit from automated support approaches, while also recognizing the irreplaceable role of human intellect in certain aspects that cannot be matched by state-of-the-art AI techniques.

Keywords: Peer Review · Cybersecurity · AI · Machine Learning · LLMs.

1 Introduction

The scientific review and decision-finding process (Fig. 1) is highly reliant on peer reviewers' judgment and agreement. The decisions ultimately depend on

† These authors contributed equally to this work.
∗ Corresponding authors.

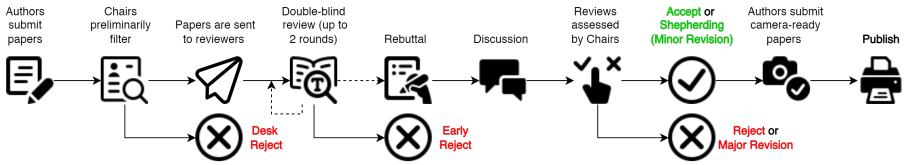


Fig. 1: The paradigm of the peer-review process for "top-tier" security conferences. This diagram includes all steps of the post-submission life cycle of a paper. Dashed lines indicate optional procedures. Bold and color fonts indicate decisions.

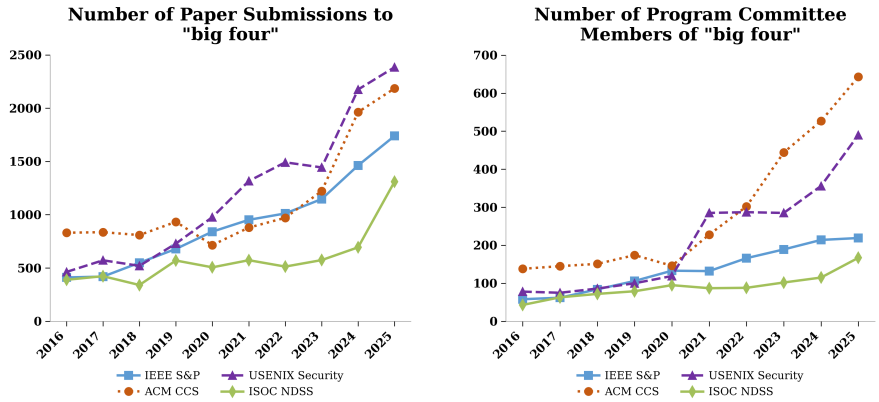
their evaluation of aspects like technical correctness, novelty, and coverage of experimental results [22], as well as more subjective aspects like creativity, applicability, and scientific contribution. Even though this process can be biased, no better approach than human inspection is known for judging scientific progress.

Advances in Artificial Intelligence (AI) and machine learning (ML) raise the question of to what extent AI models could support and complement reviewers in the scientific review process [23]. Analyzing differences between human review results and ML predictions can help uncover hidden aspects of the decision-making process. The application of ML to predict peer-review outcomes is intriguing as it challenges both the limits of AI and the logic of scientific publication. We expect that ML-based techniques can be used to predict a certain part of the review decision process, but not fully. While related approaches have been studied in other domains [30,4], to the best of our knowledge, our work is the first focusing specifically on security conferences and comparing modern Large Language Models (LLMs) against traditional models.

The broader research community is already debating how AI should be used in peer review. A 2025 *Nature* news feature reports that AI tools are increasingly involved in peer review, and that many scientists are worried about risks like low-quality or untrustworthy reviews [16]. AAAI publicly documented an AI-assisted peer review pilot for AAAI-26, where AI outputs are designed to be supplementary and are constrained by explicit safeguards [1].

Beyond individual conference pilots, dedicated venues have recently emerged to explore the broader role of AI agents in scientific discovery and evaluation. For example, the Open Conference of AI Agents for Science 2025[1] explored the future of AI-driven scientific discovery through transparent AI-authored research and AI-driven peer review. Similarly, the Conference of Synthetic Security Research[2] focuses on the use of generative and autonomous AI systems in security research, including synthetic experiments, automated vulnerability discovery, and AI-assisted analysis. These initiatives highlight that AI is not only being considered as a tool for assisting reviewers, but also as an active participant in scientific knowledge production and evaluation.

[1] https://agents4science.stanford.edu/
[2] https://synsec.org//



(a) Paper submission trends (b) Program committee trends

Fig. 2: The number of (a) paper submissions and (b) program committee members (excl. PC chairs) in the Big-4 security conferences 2016–2025.

We set out to investigate the predictability of ML on review outcomes for computer security and privacy, since this research area has seen a tremendous increase in paper submissions in the last 10 years (see Fig. 2a). This surge has challenged the peer-reviewing process at first-tier security conferences[3] and strained reviewers' ability to provide timely, comprehensive reviews. Organizers of these conferences have expanded reviewer pools (see Fig. 2b) and introduced journal-like revision opportunities, but the volume of submissions remains challenging.[4]

Beyond the rapid growth in submissions, the computer security community has several review-specific characteristics that make it a meaningful case study. First, security papers are often evaluated not only on novelty and writing clarity, but also on system realism and verifiability, and many top venues have promoted artifact evaluation and reproducibility-related practices, which further increases the reviewing workload and changes what reviewers focus on. Second, security research can be sensitive in terms of disclosure and potential misuse, so reviewers tend to be cautious about claims, evidence, and presentation. These factors suggest that studying AI-assisted decision prediction in the security peer-review setting is not only about scale, but also about domain-specific reviewing constraints and risks. Furthermore, papers under review may suffer from other issues in the reviewing process, like reviewers' personal preferences and area of expertise. Although these problems can partially be mitigated (e. g., by reviewer bidding and automated paper assignment based on reviewer expertise due to self-authored papers), adding some more objective decision elements to the review process may

[3] The "Big-4" top-tier conferences in computer security and privacy: ACM CCS, IEEE S&P, ISOC NDSS, and USENIX Security.

[4] We direct readers to Appendix A for a detailed overview of the peer-review process in the "Big-4".

have appreciated benefits. Soneji [22] conducted a qualitative study (interviews with 21 reviewers and chairs) to understand the peer-review process in computer security. Our work is complementary, as we conduct a quantitative investigation.

Related endeavors have been pursued in other areas of computer science. The computer vision community proposed methods to automatically assess paper quality based on visual features of papers [27,7]. However, that line of research relied solely on a paper's appearance, neglecting its textual content. In Natural Language Processing (NLP), researchers proposed content-based methods to grade essays [11] and score open-response answers [10], and to predict conference acceptances using text on small scales. Taghipour [25] and Alikaniotis [2] leveraged neural networks for automated essay scoring. While relevant, those works focused on educational essays rather than scientific papers, fundamentally differing from our goal. We consider a 2018 paper by Yang [30] to be the most relevant prior research: they introduced the task of automatic academic paper rating, constructing a dataset from arXiv and proposing a hierarchical CNN classifier. Their method was reported as state-of-the-art for general paper rating. However, they did not focus on computer security papers or on analyzing implications for the peer-review paradigm. Another interesting work was by Bartoli [4], who proposed a model to automatically generate plausible review text for a paper (aiming to deceive readers). They used traditional methods (to avoid the need for large training data) and focused on review text generation, not decision prediction.

Our work differs from these related efforts as follows: (1) We formulate the problem as a binary classification model predicting an academic paper's acceptance decision. (2) We focus specifically on computer security conference papers (especially the "Big-4"). (3) We incorporate and discuss state-of-the-art large language models (e.g., ChatGPT, DeepSeek, Qwen) as an AI reviewer for comparison. (4) We analyze and discuss the implications of our findings for the peer-review process in security.

In short, this paper makes the following major contributions:

1. We build a large dataset with over 25,000 papers. It consists of over 19,000 accepted conference papers and over 6,000 preprint papers collected from the arXiv preprint website.
2. We train ML models to predict whether a paper is to be accepted by top-tier conferences in computer security and privacy and show their quantitative predictive results. Our experiments show that our best models can achieve approx. 91% accuracy in predicting the reviewing decisions for security papers, significantly outperforming general-purpose LLMs by a wide margin.
3. We conduct further experiments to explore the capability of our method in dealing with abstracts and novel papers. Based on our findings, we highlight a distinct performance gap between domain-tuned ML embeddings and off-the-shelf LLMs, and discuss how vulnerabilities to temporal shifts necessitate keeping human experts in the loop for security peer reviews.
4. Our analysis reveals that while specialized ML models can predict outcomes with higher accuracy, general-purpose LLMs struggle with a strong "Accept"

bias; thus, we propose that AI is currently best positioned as an assistive triage tool rather than an independent evaluator in the peer-review process.

Open Science. To support reproducibility and future research, the source code and experimental artifacts of this work are publicly available on GitHub.[5]

2 Related Work and Background

2.1 Related Work

Several approaches have been proposed for related tasks [14]. In general, they can be grouped into (i) vision/layout-based approaches and (ii) text-based approaches (including recent LLM-based reviewing assistance).

Vision-based Approaches. Some work explores whether a paper's visual layout (e.g., figures, tables, and overall formatting) correlates with acceptance decisions. For example, Huang [7] studied "paper gestalt" using deep learning, and even explored generating paper-like layouts. These studies suggest that visual cues (such as figure/table density and layout style) may contain weak signals. However, purely vision-based models are usually unsuitable for security papers, because security submissions often have less standardized visual patterns than, for example, computer vision papers. More importantly, layout features ignore the actual technical content, while real peer review heavily depends on content quality (e.g., novelty, correctness, and clarity). For this reason, our approach focuses on textual content rather than layout features.

Text-based Approaches. NLP researchers have explored content-based analysis to assist in reviewing and scoring. Taghipour [25] proposed neural models (CNN and LSTM) for automated essay scoring and showed improvements over earlier methods on essay datasets. Alikaniotis [2] similarly used LSTM-based models for essay scoring. Although related, these works focus on educational essays rather than scientific papers.

For scientific papers, Yang [30] studied automatic academic paper rating by building a dataset from arXiv papers and training a modular hierarchical CNN, but the reported performance was limited, and the setup was not specialized for security papers. A closely related and widely used resource is *PeerRead* [9], which provides paper texts, review-related metadata, and (for some venues) accept/reject outcomes; it has enabled many studies on modeling peer review signals and predicting acceptance decisions. Compared with these works, our study targets the security reviewing setting, and we also explicitly evaluate the role of a strong LLM assistant (e.g., ChatGPT) in decision prediction.

A related line of work focuses on generating review reports automatically. Bartoli [4] showed that it is possible to generate plausible-looking paper reviews with template-based or lightweight methods. Our work differs in goal: instead of generating review text, we focus on predicting review decisions for security papers.

[5] https://github.com/ethanyuan-research/Assessing-AI-Performance-in-Cybersecurity-Peer-Review

LLMs and Peer Review. With the rise of large language models, there has been increasing discussion and early empirical exploration of using LLMs in academic workflows [8,24]. More recently, work has started to systematically study LLMs in the peer-review process, such as their potential roles, limitations, and risks (e.g., hallucinations and bias) [13]. In addition, a 2025 *Nature* news feature reports that AI tools are increasingly involved in peer review and that many scientists are worried about the reliability and integrity of AI-influenced reviews [16]. Empirically, Latona et al. [21] studied the prevalence and potential impact of AI-assisted peer reviews at ICLR 2024, showing that AI assistance in reviews is not just a theoretical possibility but is already present in practice. At the process level, AAAI documented an AI-assisted peer review pilot program for AAAI-26 and emphasized safeguards such as no AI-generated scores or recommendations and that final decisions remain fully human [1]. These developments motivate careful evaluation of how much LLM outputs can be trusted for review-related tasks, which aligns with our goal to assess predictability in security peer review.

Besides the above technical works, an IEEE S&P paper by Soneji [22] provides important inspiration. They performed a qualitative study of the security peer-review process via interviews, and many participants expressed negative sentiments about the current system. These findings motivate us to explore the predictability of peer-review outcomes for security papers quantitatively.

Beyond reviewing, LLMs are also being used in other parts of the research workflow. For example, Project Rachel studied AI authorship by creating and tracking a complete AI academic identity and observing the scholarly ecosystem's response [15].

2.2 Background

Doc2Vec. Doc2Vec [12] represents each document as a vector, extending the Word2Vec idea to document-level embeddings. It is an unsupervised method that benefits from large-scale corpora. We leverage this property by training Doc2Vec on a large set of academic papers, so that it can produce useful document embeddings for our ML-based reviewing pipeline.

LLMs. Large Language Models (LLMs), exemplified by ChatGPT [17], DeepSeek, and Qwen, represent a major change in natural language processing. Built on transformer architectures and trained on large corpora, these models can understand and generate coherent text across many tasks. Prior work suggests that strong LLMs (e.g., GPT-4) show early signs of generalized reasoning ability, although it is far from a complete AGI [6]. As a result, the adoption of LLMs in research and writing has triggered active discussion in the community [8,24]. In addition, Project Rachel studied AI authorship by creating and tracking a complete AI academic identity, highlighting that AI is also affecting scholarly writing and the publication ecosystem [15].

Table 1: The two subsets of papers used in our approach.

Subset	Component	Label	Venue	Amount	Usage
Proceedings	Accepted papers	Accept	Top-tier computer science conferences	19,028	Doc2Vec + Classification (Big-4 papers); only Doc2Vec (others)
Preprints	arXiv preprints	Reject	Drafts from security conferences and other security preprints	6,625	Doc2Vec + Classification

3 Data Collection

To train and test the paper classification models, we needed a large number of scientific papers. Since no existing dataset could be utilized directly for our task, we first created such a dataset.

Given the unsupervised nature of Doc2Vec (see Section 4.2), its performance improves with larger training data. However, the limited number of security conference papers alone would impede training. To overcome this, we gathered accepted papers from prominent computer science conferences outside the security domain. These conferences span various topics and share substantial vocabulary with security papers, making them suitable for pre-training the Doc2Vec model. To generate high-quality embeddings for security papers, we adopted a transfer-learning approach: our Doc2Vec model is initially trained on general computer science papers, then fine-tuned on computer security papers.

To build the dataset, we collected published papers from first-tier computer science conferences in domains such as computer vision, networking, and security. We also incorporate papers from the arXiv preprint repository as negative (rejected) samples; details are explained in the following subsection.

3.1 Dataset Composition

Our dataset comprises two subsets: *Proceedings* and *Preprints* (Table 1).

– The **proceedings subset** consists of over 19,000 *published* papers from top computer science conferences, including both security-specific venues and broader CS domains. Detailed stats on venues and sample sizes are in Table 5 (Appendix).
– The **preprints subset** comprises papers obtained from the arXiv preprint site. This subset plays a crucial role in the second stage of training, where our models aim to predict acceptance or rejection by the "Big-4" conferences. We use the arXiv papers as negative (rejected) samples, guided by heuristic selection rules (outlined below). This approach approximates real review outcomes from actual security conferences, since directly obtaining such data is challenging both logistically and ethically: (1) Accessing review decisions is difficult due to the closed, confidential nature of the process, and (2) Collecting papers along with their outcomes without consent would be inappropriate.

Thus, we focus on public data sources, and after careful consideration, we identified arXiv as a viable choice. Widely used in CS, arXiv provides a large volume of samples and a research-friendly usage policy, avoiding ethical issues.

We acknowledge that this approach has limitations. Extracting likely-rejected papers from arXiv can introduce errors and only approximate actual outcomes. Nonetheless, we argue that our heuristic selection rules reflect reality sufficiently well.

In particular, we treat published Big-4 papers as positive samples, and select negative samples from arXiv security papers using the following rules:

– **Rule 1**: For arXiv preprint papers that eventually get published at one of the "Big-4" conferences, if a paper has multiple versions on arXiv and its *first* version is at least one year older than its *last* version, we consider the first version as a negative sample. This rule is based on the (simplifying) reasoning that if a paper was accepted by the Big-4 after at least a year of revisions, its first version was likely already decent but not of sufficient quality for the Big-4 at that time, making it a suitable negative sample. Considering that content improvements take time, we chose one year as a heuristic threshold based on our experience in refining papers. This scenario is relatively rare in our data, producing only 158 negative samples (whose corresponding final versions are included in the positive set).

– **Rule 2**: For arXiv preprints that are eventually published in peer-reviewed security-related venues outside the Big-4[6], we use their first arXiv versions as hard negative samples with respect to Big-4 acceptance. This rule is not intended as a judgment of paper quality or venue quality. Rather, it provides a practical proxy for papers that are relevant to security research but outside the positive Big-4 target class.

– **Rule 3**: If an arXiv preprint paper was not published at a peer-reviewed venue and it was first submitted prior to 2021, we consider it a negative sample. This rule serves as a fallback to ensure we gather enough negative samples when the first two rules do not yield sufficient data. The reasoning is that preprint-only security papers (especially those posted long ago) were likely submissions to Big-4 or other reputable security conferences that did not get accepted. We limit this to preprints from before 2021, as more recent papers might still be under review. We note, however, that Rule 3 has a caveat: since it contributes the majority of our negative samples, it likely contains many papers whose quality is substantially below the Big-4 threshold. We therefore carefully distinguish its role from the stronger negative samples in Rules 1 and 2; Rule 3 primarily functions as a "catch-all rule" to complete the scale required for training.

Applying these rules, we get a hierarchical set of negative samples: Rule 1 yielded 158 samples, Rule 2 yielded 713 samples, and Rule 3 yielded 5,754 samples. Rule 3 provided the amount of samples needed to train a deep learning model with a massive dataset like Doc2Vec, while the samples from Rules 1 and 2 served as "hard negatives" – these papers were of a quality close to the acceptance standards of top conferences. This combination enabled our model to learn robust semantic features that captured both general rejection patterns

[6] Those security conferences considered for Rule 2 are: ESORICS, RAID, ACSAC, DSN, CSFW, ASIACRYPT, SecureComm, AsiaCCS, ACNS, SAC, ACISP, ICICS, ICISC, ISC, FSE, WiSec, SEC, SACMAT, CT-RSA, DIMVA, and ISPEC.

and fine-grained quality differences. The final 6,625 negative samples formed a balanced distribution with the collection of papers accepted by the four major conferences (see Table 5), thus preventing the classifier from being biased towards the majority class.

Ultimately, although these heuristic labels contain noise, they enable us to construct, to our knowledge, one of the earliest large-scale datasets of predicted peer review results for security conferences. This dataset may serve as a starting point for subsequent research into stronger negative sample construction methods and label denoising techniques.

3.2 Data Acquisition

All papers in the proceedings subset were obtained from public sources. We developed automated crawlers to retrieve accepted papers from various conference websites, supplementing with manual downloads for any missing papers.

For the preprints subset, we first fetched arXiv metadata from Kaggle[7] and then used a crawler to download all cs.CR (cryptography and security) papers from `export.arxiv.org`. We then used the DBLP[8] database to check the publication venue of each downloaded paper, applying our heuristic rules to select negative samples. DBLP provides bibliographic information on major CS publications and has a robust process for quality-checking entries [26]. Its search function, powered by CompleteSearch [5], was tested and found to reliably retrieve publication information even if a paper's title changed slightly over time.

4 Methodology

We first present our terminology (Sec. 4.1) before defining the fundamentals of our vector embedding approach (Sec. 4.2). We then detail our paper classification pipeline (Sec. 4.3, Fig. 3) and outline our methodology for the LLM experiments (Sec. 4.4).

4.1 Terminology

The reviewing process can be viewed as a text classification problem, formally defined here. We collected a set of papers $\mathbb{D} = d_1, \ldots, d_N$, where $N = |\mathbb{D}|$. A paper d_i consists of M words ($d_i = w_1, \ldots, w_M$). The raw papers are mapped to a high-dimensional vector space by a Doc2Vec model, representing each paper as a vector $\boldsymbol{d_i}$.

Definition. *The binary classifier, denoted $f(\cdot)$, is the function defined by*

$$f(\boldsymbol{d_i}) = \begin{cases} 1, if\ \eta(\boldsymbol{d_i}) > 1/2 \\ 0, \quad otherwise, \end{cases} \tag{1}$$

[7] `https://www.kaggle.com/Cornell-University/arxiv`
[8] `https://dblp.org/`

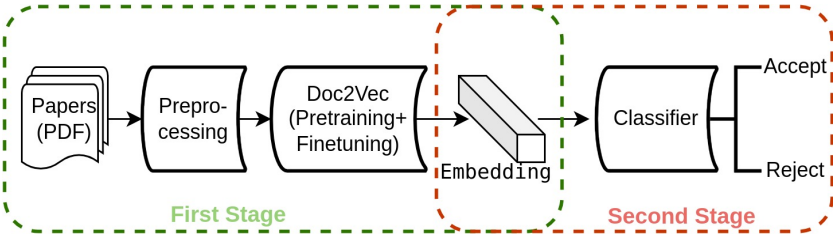


Fig. 3: The pipeline of our paper classification model. Left: Doc2Vec generates document embeddings from papers. Right: classification models predict Accept/ Reject based on the embeddings.

where $\eta(\cdot)$ is an ML classification algorithm that outputs a probability score in $[0, 1]$. A prediction of 1 means the classifier predicts the paper should be accepted, while 0 indicates rejection.

4.2 Problem Definition: From Papers to Vector Embeddings

Here we briefly outline how Doc2Vec transforms a paper into a high-dimensional vector. In the Doc2Vec framework, each raw paper is projected to a unique document vector [12]. Let $\mathbf{d}_i \in \mathbb{R}^n$ denote the embedding of paper d_i and stack all document vectors into

$$\mathbf{D} = \left[\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_N\right] \in \mathbb{R}^{n \times N}, \tag{2}$$

where each column corresponds to one paper. Similarly, each word in the vocabulary is mapped to a unique vector during training, stacked as columns in $\mathbf{W} \in \mathbb{R}^{n \times |V|}$.

Given a paper d_i and a sequence of words $w_1, \ldots, w_M$ in that paper, the Doc2Vec objective maximizes the average log-likelihood of predicting the center word from its surrounding context and the document vector:

$$\max_{\theta} \; J(\theta) = \frac{1}{M-2k} \sum_{t=k+1}^{M-k} \log p(w_t \mid d_i,\; w_{t-k}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+k}; \theta)\,, \tag{3}$$

where k is the context window size and θ denotes the model parameters. The document vector $\mathbf{d}_i$ acts as a memory for the document's topic, capturing information not fully represented by the local context. If we write training as a minimization problem, we minimize $-J(\theta)$.

The conditional probability $p(\cdot)$ is commonly parameterized by a softmax over the vocabulary. Let $\mathbf{y} \in \mathbb{R}^{|V|}$ be the logit vector produced from the combined representation of the context words (via $\mathbf{W}$) and the document vector (via $\mathbf{D}$). Then

$$p(w_t = j \mid d_i, \text{context}) = \frac{\exp(y_j)}{\sum_{u=1}^{|V|} \exp(y_u)}, \tag{4}$$

and in particular,

$$p(w_t \mid d_i, \text{context}) = \frac{\exp(y_{w_t})}{\sum_{u=1}^{|V|} \exp(y_u)}. \tag{5}$$

The function that forms the input representation (e.g., concatenation or averaging of context word vectors together with $\mathbf{d}_i$) is implementation-dependent; we omit it for brevity.

Training aims to find the θ that maximizes $J(\theta)$; we use stochastic gradient descent (via backpropagation) to update the document and word vectors and other network parameters. After training, we perform an inference step to compute a document vector for any new paper (by freezing the learned model and feeding in the new text). The resulting document embedding is then fed into machine-learning classifiers (e.g., Naïve Bayes, Logistic Regression, SVM, KNN) to predict whether the paper should be accepted or not. Finally, these classifiers output a prediction for each paper.

4.3 Method: From Preprocessing to Classification

Our methodology comprises three main stages (see Figure 3):

1. **Preprocessing:** First, PDF files are converted to plain text, after which anonymization is performed by removing author names and affiliations. Subsequently, the corpus is normalized (e.g., removal of LaTeX and formatting artifacts, page headers, etc.).
2. **Document embedding:** Next, we turn the normalized corpus into document embeddings (latent representations) using a trained Doc2Vec model.
3. **Classification:** Finally, we use classification algorithms to predict whether each paper will be accepted or rejected.

Preprocessing of Collected Data: The collected papers are processed into the proper form to ensure their compatibility with subsequent modeling steps.

- *PDF to Text.* Since all the collected papers are in PDF format and PDF files contain not only content but also layout and typesetting information, they need to be transformed into plain text to fit into NLP models like Doc2Vec in the next stage of processing. The tool used for such a task is the open source software PDFMiner. PDFMiner takes PDF documents as input, parses and analyzes them, and outputs plain text documents.
- *Text Normalization.* Following the conversion of PDF to plain text, we normalize the text to enhance efficiency during Doc2Vec training. The normalization techniques that we use in the preprocessing include: Contraction Expansion, Lemmatization and Stemming, Stop Word Removal, etc. We refer the readers to Appendix B for a more detailed explanation of these techniques. In addition to conventional text normalization techniques, we employ task-specific methods to process and normalize the corpus. This involves removing extraneous content such as author information, bibliography, and publication-related details from the documents. These techniques ensure that our pipeline is double-blinded just like the double-blind peer-review process,

which is different from previous NLP-based researches that often take account of the authors.

Document Embedding: Different methods can be used to generate document embeddings (Section 2). Our choice is to use Doc2Vec for the proposed automatic paper reviewing pipeline because it is simple yet powerful, unsupervised, and able to get better as the training set becomes larger. To enhance the embedding generation capability of the Doc2Vec model for computer security papers, a two-phase training approach is employed. In the initial pretraining phase, the model is trained on a comprehensive dataset consisting of both security and non-security computer science papers, utilizing papers from both subsets. Subsequently, in the second phase, the model is fine-tuned exclusively on security papers to enhance its sensitivity to the security domain.

Classification: In our experiments, 14 ML classification algorithms and 2 ensemble learning methods are tested. We report their results along with our discussion. Finally, we work with 7229 positive samples (all are published "Big-4" papers) and 6625 negative samples (arXiv preprints selected by heuristic rules).

Using Abstracts for Prediction: One interesting question is how well one can predict outcomes using only the paper's abstract. To explore this, we extract each paper's Abstract section (or the first 2,000 characters if no explicit abstract is found) and train a Doc2Vec model specifically on abstracts to generate embeddings. The abstract text is then normalized and used for training and testing under the same classification pipeline.

4.4 LLM-based Method

To comprehensively evaluate the capabilities of Large Language Models (LLMs) in automated academic paper reviewing, we orchestrated a series of rigorous empirical experiments leveraging three state-of-the-art LLM systems: ChatGPT, DeepSeek, and Qwen.. These models were strategically selected to represent leading LLM paradigms from different ecosystems, including both widely adopted global models and strong open-source or regionally influential models, enabling a more comprehensive and diverse evaluation. We evaluate these three representative LLMs with varying knowledge cutoffs: GPT-4o mini (Oct. 2023), Qwen2.5-72B-Instruct (Oct. 2023), and DeepSeek-V3 (July. 2024). To ensure rigorous evaluation, we limited our test data to papers published in 2025. Since these papers were published much later than the training data cutoff date for all the LLMs being evaluated, this setting guarantees that the models must predict results from unseen papers, rather than simply regurgitating reviews known from the training set. Specifically, 934 published papers from CCS '25, NDSS '25, and USENIX Security '25 are used as positive samples, and 934 papers from the arXiv dataset are used as negative samples for testing.

Following the zero-shot role-play prompting paradigm, we framed the task by instructing the LLMs to operate under the persona of an expert academic evaluator. The exact, uninhibited prompt template delivered to the language models is formulated as follows:

```
You are an experienced and fair reviewer from top cybersecurity
conferences (NDSS, IEEE S&P, CCS and USENIX Security). I will give
you a complete research paper to read and review. Please read the
full text carefully. I want you to decide whether this paper should be
accepted or not. You must first tell me your decision with ‘‘Accept’’ or
‘‘Reject’’, and then explain your reasons in concise language.
```

5 Experimental Evaluation

We conduct experiments to evaluate the predictive performance of the proposed method and to report the results on three LLMs.

5.1 Experimental Results

As a baseline for comparison, we assume an oracle that randomly decides to accept or reject without bias. Such a baseline would achieve 50% accuracy on average. We compare various ML models to this Random Guess baseline. All classification models were trained on 80% of the dataset and evaluated on the remaining 20%.

We tested 14 different classification algorithms, as well as two model ensemble methods (Voting and Stacking). The final classification results for each algorithm are shown in Table 2. Across all classifiers, the average accuracy on the testing data is 85.98% and the median accuracy is 88.95%. The average F1 score is 0.8553 and the median F1 score is 0.8900. Moreover, most classifiers train and test in under 40 seconds, and inference time for all models is negligible compared to training time.

5.2 Results of Model Ensemble

Ensemble learning uses a set of models to achieve better performance than any single model [19,20]. By allowing models to "learn from each other's strengths", the ensemble can improve generalization. We selected five diverse, high-performing models (Logistic Regression, RBF SVM, Gaussian Process, AdaBoost, and LDA) from the 14 tested models, and applied "Voting" [28] and "Stacking" [29] ensemble strategies. The final estimator in our Stacking ensemble is a decision tree classifier.

The ensemble results correspond to the "Voting Classifier" and "Stacking Classifier" rows in Table 2. Both ensemble models achieved accuracy and F1 above 0.85. In particular, the Voting strategy reached 0.91 accuracy, matching the best individual models. In theory, an ensemble is usually more robust than a single classifier. Therefore, for further discussion we use the Voting classifier as representative of our results. Generally, the accuracies and F1-scores of all models are fairly close to each other, indicating that the false negative and false positive rates are similar. For example, the "Voting Classifier" yielded 92 false positives and 157 false negatives.

Table 2: **Experimental results.** Time is training+testing time.

Algorithm	**Accuracy**	**F1 Score**	**AUC**	**Time (s)**
Random Guess	0.5	0.5	0.5	-
Voting on Abstract	0.825	0.825	-	780.85
Voting Classifier	0.910	0.910	-	1810.63
Stacking Classifier	0.866	0.866	0.866	6467.95
Logistic Regression	0.904	0.904	0.965	0.14
Linear SVM	0.908	0.908	-	35.75
RBF SVM	0.898	0.897	-	27.20
Poly SVM	0.888	0.889	-	15.60
Gaussian Process	0.908	0.907	0.966	1726.22
Decision Tree	0.747	0.747	0.747	4.91
Random Forest	0.865	0.864	0.942	8.34
Boosted Tree	0.879	0.879	0.944	264.2
MLP	0.895	0.894	0.959	4.06
AdaBoost	0.891	0.891	0.949	0.63
Gaussian Naive Bayes	0.846	0.844	0.896	0.15
KNN	0.599	0.536	0.829	0.22
LDA	0.909	0.909	0.966	0.43
QDA	0.843	0.840	0.887	0.62
Average	0.8598	0.8553	0.9097	647.94
Median	0.8895	0.8900	0.9430	6.63

We report the abstract-only prediction result in the "Voting on Abstract" row of Table 2. Using the Voting classifier, we achieve an accuracy of 82.5% and F1 score of 0.825. Compared to using the full paper, performance drops when relying only on the abstract, which is expected. An abstract provides a high-level summary of the core idea, methodology, and results (e.g., whether it achieves state-of-the-art performance). Thus, the abstract is a strong indicator of the paper's contribution and merits, which are key acceptance criteria. Despite the performance loss, predicting outcomes from abstracts alone still far exceeds the Random Guess baseline, and it is notably faster to use abstracts than full papers.

5.3 Results of LLMs

We utilized the official APIs of three representative language learning models (i. e., ChatGPT, DeepSeek, and Qwen) and performed the thesis evaluation task using two input methods: the full text of papers and only the abstracts. During testing, some API requests failed, or LLM produced answers not conforming to our prompt (e.g., refusing to decide). We discarded such cases and focused on valid outputs. The results, shown in Table 3, indicate that the accuracy of LLMs is

Table 3: **LLMs Results.** The "Answer" column shows the valid responses obtained. "Accept" and "Reject" columns show how many Accept/Reject predictions LLMs made.

Input	Answers	Correct Predictions	Accuracy	Accept	Reject	Accuracy of Big-4	Accuracy of arXiv
ChatGPT-Abstract	1866	1021	54.7%	1708	158	96.2%	13.2%
ChatGPT-Full text	1834	991	54.0%	1728	106	97.4%	9.1%
DeepSeek-Abstract	1868	1200	64.2%	1088	780	72.5%	56.0%
DeepSeek-Full text	1866	1147	61.5%	1620	246	98.3%	24.7%
Qwen-Abstract	1862	1039	55.8%	1437	425	82.9%	28.6%
Qwen-Full text	1867	1146	61.4%	1514	353	92.5%	30.3%

only about as good as random guessing, regardless of input type. Notably, LLMs tended to predict "Accept" for almost every paper. We discuss these findings further in Section 6.

In contrast, under this minimal configuration, LLMs only achieve about 60% accuracy on our task. These results demonstrate the basic effectiveness of the Doc2Vec-based method. More specifically, all SVM-based methods (with various kernels) achieved high accuracy with a reasonable amount of training data, and the simple linear SVM performed the best among individual models.LLMs, by contrast, showed a strong bias towards predicting "Accept" for almost every paper, which severely limited its accuracy. We further discuss the tendencies and limitations of LLMs in the next section.

6 Discussion

Our experimental results offer several insights into predicting peer-review outcomes for security papers, as well as the strengths and limitations of current AI models in this context.

Accuracy of ML Predictions. Our Doc2Vec-based classification pipeline achieved around 90% accuracy in predicting acceptance decisions on a balanced dataset of security conference papers. This suggests that, to a significant extent, there are observable textual patterns or features correlated with paper acceptance. For example, accepted papers might have clearer writing, more complete experiments, or other qualities that our model can capture as part of the embedding. Importantly, the model's errors are roughly evenly split between false positives and false negatives, indicating that it does not exhibit a strong bias toward predicting either acceptance or rejection. For instance, the Voting ensemble classifier produced 157 false positives and 92 false negatives, which is reasonably balanced. This balanced error behavior is desirable, as an unbalanced model (e.g., always predicting rejection) would be less useful in practice.

Impact of Using Abstracts Only. Our additional experiment using only abstracts as input shows a non-trivial drop in accuracy (from 91% to 82.5%

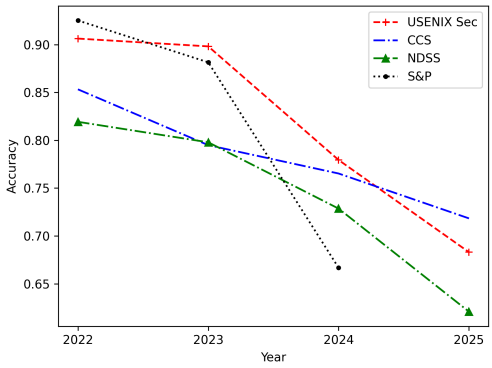


Fig. 4: Model performance under temporal distribution shift (train: pre-2022; test: 2022–2025).

with the Voting classifier). This drop confirms that information beyond the abstract — such as technical details, experimental depth, and perhaps writing quality throughout the paper — contributes to the model's predictive power. The abstract does encapsulate a paper's key idea and contributions, which is why abstract-only predictions were still far above chance. However, important signals that reviewers consider (e.g., whether the methodology is sound, whether results are convincing) often require reading beyond the abstract. Thus, while an abstract gives a strong hint (and using it alone may be useful for quick screening), it cannot fully substitute for the entire content in prediction tasks.

Ability to Handle Temporal Distribution Shift. A common concern about ML-supported review is that algorithms may struggle when faced with papers that differ substantially from historical data. Since our model is trained on past acceptance decisions, it may fail to generalize when research topics, methodologies, or trends evolve over time. To investigate this issue, we conducted a temporal generalization experiment. Specifically, we trained our classifier using papers published prior to 2022 and evaluated it on papers from 2022–2025. This setup simulates a scenario where the model is unaware of more recent research developments.

As shown in Fig. 4, the model's performance decreases substantially over time. For example, accuracy on USENIX Security drops from over 90% in 2022 to approximately 68% in 2025, and accuracy on NDSS declines to around 62%. This performance degradation suggests that temporal distribution shift — potentially caused by emerging topics, new methodologies, or evolving evaluation standards — significantly affects model reliability. We note that this experiment does not directly measure novelty in a formal sense; rather, it uses temporal separation as a proxy for evolving research trends. Future work could incorporate explicit novelty or out-of-distribution detection techniques (e. g., [18,3]) to more rigorously assess this issue.

Limitations and Biases. Our approach has some clear limitations: (1) It uses only textual information from papers and past acceptance decisions. It cannot evaluate a paper's technical correctness or deep originality – factors that require human expertise. The model could pick up on superficial proxies (e.g., presence of certain buzzwords, writing clarity, or paper length) that correlate with acceptance, without truly understanding the content. (2) Since it learns from past data, it may inherit biases present in past decisions. For example, if historically certain subtopics or methodologies were favored (or disfavored) by reviewers, the model might internalize those preferences. This raises the possibility that such a model could perpetuate or even amplify existing biases (for instance, undervaluing papers on unconventional topics). We partially mitigated this by anonymizing content (removing author names and affiliations) to prevent trivial biases (like famous author effect), but subtler biases could remain. Our error analysis did not reveal strong biases – the model seems to mispredict somewhat evenly across different types of papers – but a more thorough bias audit would be necessary before any practical deployment. (3) A notable limitation of our LLM evaluation is its reliance on basic zero-shot prompting. It does not explore task-specific model fine-tuning or advanced prompt engineering techniques, such as Chain-of-Thought (CoT) reasoning, few-shot learning, or explicit rubric design. Furthermore, our evaluation utilized single-pass prompts without multi-step reasoning or agentic frameworks. As a result, the observed 60% accuracy should be interpreted as a baseline measurement of these unoptimized zero-shot capabilities, rather than the full potential of LLM-based approaches. (4) Newer versions of the LLM models may yield different results. Future research optimizing LLMs for peer review through these advanced methodologies may lead to significant improvements. (5) Finally, while LLMs can generate explanatory rationales for academic paper decisions, their reliability will also be undermined by ungrounded fabrications and conflicting predictions. These limitations stem from a lack of domain-specific knowledge, an inherent positivity bias, and a struggle with long-form reasoning over complex documents. Consequently, the models can produce inconsistent rationales, potentially sometimes providing contradictory acceptance and rejection recommendations for the same paper.

Potential Use Cases. Given its limitations, our AI is not meant to replace human reviewers. However, it could serve as a support tool in the peer-review process. One potential use case is as a triage system: it could flag submissions that are very likely rejects (or very likely accepts) to help program committees prioritize their reviewing resources. For example, papers that the model predicts with high confidence to be accepted might be routed to senior reviewers for confirmation, whereas those predicted to be rejected could be given an extra look to ensure no novel gem is being overlooked. The model could also be used to assist new reviewers by highlighting sections of a paper that contributed most to the model's decision (using explainability techniques), thereby pointing out possibly important factors. Of course, any such use must be done carefully, keeping humans in the loop and aware of the model's fallibility.

Ethical Considerations. Automating aspects of peer review raises ethical issues. Authors might fear that an algorithm could unfairly judge their work or be biased. It is crucial that any AI assistance in peer review be used transparently and not as the sole arbiter of decisions. Our study is an exploration of what current AI can (and cannot) do; it is not an endorsement to replace program committees with algorithms. We believe the insights gained (e.g., the textual factors correlated with acceptance) could even be shared with authors to help them improve their papers, rather than being hidden in a secret decision-making tool.

7 Conclusion

To study the predictability of peer-review outcomes for security papers, we evaluated a Doc2Vec-based classification approach and compared it against large language models on top-tier security conference submissions. The Doc2Vec-based method achieved approximately a peak prediction accuracy of approximately 91%, whereas LLMs reached only about 60% accuracy in a zero-shot baseline setup. While our method exhibits reasonably high accuracy in predicting review decisions for security research papers, it still has a non-negligible error rate. Meanwhile, utilizing advanced AI tools like ChatGPT for academic review still requires substantial advancements. We conclude that our proposed method can predict review outcomes with respectable accuracy, but the error rate remains significant. Nevertheless, AI models cannot replace human reviewers. Our model learns patterns from historical decisions but cannot judge a paper's novelty or technical correctness, and it may even reinforce biases present in the training data. Therefore, such tools should serve only as assistive aids (e.g., to help triage submissions), not as final decision-makers in the peer-review process. While the peer-review system has historically adapted to various challenges, future research should build on our study by testing AI capabilities on a more challenging and realistic task: distinguishing between polished, rigorously reviewed papers from top-tier "Big-4" conferences and strong tier-2 security venues (e.g., ACSAC, ESORICS, RAID, DSN, etc.).

Acknowledgements

This work was supported by the Center for Cyber Security at New York University Abu Dhabi (NYUAD). This research was carried out on the High Performance Computing resources at NYUAD.

References

1. AAAI: Faq for the AI-assisted peer-review process pilot program. PDF document (2025), https://aaai.org/wp-content/uploads/2025/08/FAQ-for-the-AI-Assisted-Peer-Review-Process-Pilot-Program.pdf, aAAI-26 AI-assisted peer review pilot documentation (accessed 2026-02-12)

2. Alikaniotis, D., Yannakoudakis, H., Rei, M.: Automatic text scoring using neural networks. In: Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers). pp. 715–725 (2016)
3. Amplayo, R.K., Hong, S., Song, M.: Network-based approach to detect novelty of scholarly literature. Information Sciences **422**, 542–557 (2018)
4. Bartoli, A., De Lorenzo, A., Medvet, E., Tarlao, F.: Your paper has been accepted, rejected, or whatever: Automatic generation of scientific paper reviews. In: International Conference on Availability, Reliability, and Security. pp. 19–28. Springer (2016)
5. Bast, H., Weber, I.: The completesearch engine: Interactive, efficient, and towards ir & db integration. In: Third Biennial Conference on Innovative Data Systems. pp. 88–95 (2007)
6. Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y.T., Li, Y., Lundberg, S., Nori, H., Palangi, H., Ribeiro, M.T., Zhang, Y.: Sparks of artificial general intelligence: Early experiments with gpt-4 (March 2023), https://www.microsoft.com/en-us/research/publication/sparks-of-artificial-general-intelligence-early-experiments-with-gpt-4/
7. Huang, J.B.: Deep paper gestalt. arXiv preprint arXiv:1812.08775 (2018)
8. Hutson, M.: Could AI help you to write your next paper? https://www.nature.com/articles/d41586-022-03479-w (2022)
9. Kang, D., Ammar, W., Dalvi, B., Van Zuylen, M., Kohlmeier, S., Hovy, E., Schwartz, R.: A dataset of peer reviews (peerread): Collection, insights and nlp applications. In: Proceedings of the 2018 conference of the North American chapter of the Association for Computational Linguistics: Human language technologies, volume 1 (long papers). pp. 1647–1661 (2018)
10. Lan, A.S., Vats, D., Waters, A.E., Baraniuk, R.G.: Mathematical language processing: Automatic grading and feedback for open response mathematical questions. In: Proceedings of the Second (2015) ACM Conference on Learning @ Scale. pp. 167–176. ACM (2015)
11. Larkey, L.S.: Automatic essay grading using text categorization techniques. In: SIGIR. vol. 98, pp. 90–95 (1998)
12. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International conference on machine learning. pp. 1188–1196 (2014)
13. Lee, J., Lee, J., Yoo, J.J.: The role of large language models in the peer-review process: opportunities and challenges for medical journal reviewers and editors. Journal of Educational Evaluation for Health Professions **22** (2025)
14. Lin, J., Song, J., Zhou, Z., Chen, Y., Shi, X.: Automated scholarly paper review: Concepts, technologies, and challenges (2023)
15. Monperrus, M., Baudry, B., Vidal, C.: Project rachel: Can an AI become a scholarly author? arXiv preprint arXiv:2511.14819 (2025). https://doi.org/10.48550/arXiv.2511.14819
16. Naddaf, M.: Ai is transforming peer review — and many scientists are worried. Nature **639**(8056), 852–854 (Mar 2025). https://doi.org/10.1038/d41586-025-00894-7
17. OpenAI: ChatGPT. https://openai.com/blog/chatgpt (2022)
18. Pimentel, M.A., Clifton, D.A., Clifton, L., Tarassenko, L.: A review of novelty detection. Signal processing **99**, 215–249 (2014)
19. Polikar, R.: Ensemble based systems in decision making. IEEE Circuits and systems magazine **6**(3), 21–45 (2006)
20. Rokach, L.: Ensemble-based classifiers. Artificial intelligence review **33**(1-2), 1–39 (2010)

Table 4: Summary of the current review outcomes for the "Big-4" conferences.

Conference	Final Review Outcomes
ACM CCS	Accept/Minor Revision/Reject
IEEE S&P	Accept/Conditional Accept/Major Revision/Reject
ISOC NDSS	Accept/Minor Revision/Major Revision/Reject
USENIX Security	Accept/Accept on Shepherd Approval/Reject

21. Russo, G., Horta Ribeiro, M., Davidson, T.R., Veselovsky, V., West, R.: The ai review lottery: Widespread ai-assisted peer reviews boost paper scores and acceptance rates. Proceedings of the ACM on Human-Computer Interaction **9**(7), 1–28 (2025)
22. Soneji, A., Kokulu, F.B., Rubio-Medrano, C.E., Bao, T., Wang, R., Shoshitaishvili, Y., Doupé, A.: Flawed, but like democracy we don't have a better system: The experts' insights on the peer review process of evaluating security papers. In: 2022 IEEE Symposium on Security and Privacy (SP) (2022)
23. Srivastava, M.: A day in the life of chatgpt as an academic reviewer: Investigating the potential of large language model for scientific literature review. OSF preprints (2023)
24. Stokel-Walker, C.: What ChatGPT and generative AI mean for science. https://www.nature.com/articles/d41586-023-00340-6 (2023)
25. Taghipour, K., Ng, H.T.: A neural approach to automated essay scoring. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 1882–1891 (2016)
26. dblp team: How complete is dblp? https://dblp.org/faq/23593238.html (2025), accessed: 2025-11-19
27. Von Bearnensquash, C.: Paper gestalt. Secret Proceedings of Computer Vision and Pattern Recognition (CVPR) (2010)
28. Witten, I.H., Frank, E.: Data mining: practical machine learning tools and techniques with java implementations. Acm Sigmod Record **31**(1), 76–77 (2002)
29. Wolpert, D.H.: Stacked generalization. Neural networks **5**(2), 241–259 (1992)
30. Yang, P., Sun, X., Li, W., Ma, S.: Automatic academic paper rating based on modularized hierarchical convolutional neural network. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers (2018)

A Appendix: Peer-Review Process in Security

Many security conferences have turned to a more journal-style peer-review process in recent years. The final decision now includes not only accept and reject, but also revision (Table 4). Additionally, the number of review cycles have stabilized to two cycles per year: CCS and NDSS now have two review cycles per year; USENIX Security has also reduced earlier three to now two review cycles, and IEEE S&P has recalibrated to two cycles from an earlier per-month review cycle with 12 submission deadlines per year. The review is double-blind across all the review stages, meaning the identities of authors and reviewers are hidden from

each other. The review process at the "Big-4" happens in multiple review rounds; papers that make the first round get additional reviewer assignments.

B Detailed Introduction to the Preprocessing

Here we provide a more detailed introduction to the techniques we used in preprocessing to normalize the corpus:

– **Contraction Expansion**. Contractions in English are shortened word-phrase like "don't", "Y'all", "can't" and so on. Contraction expansion aims to expand contraction as their unshortened normal form, so "don't" becomes "do not", "Y'all" becomes "You all", etc.
– **Accented Character Removal**. Sometimes accented characters may appear in an English article or paper, e. g. 'ò'. In normalization, we want to delete the accent so that 'ò' becomes 'o'.
– **Lower Case** Turning all words into their lower case form could fix the problem that the model treats "You" and "you" differently.
– **Lemmatization and Stemming**. Both techniques aim to reduce inflectional forms of English words and turn them into their common base form, e. g., "are, is, am" all become "be". The difference between lemmatization and stemming is that lemmatization tries to return the dictionary form of words (lemma) using morphological analysis while stemming only attempts to delete the ends of words (affixes) and return the remained part (stem). So we applied lemmatization instead of stemming in our normalization pipeline.
– **Stop Word Removal**. Stop words are words that are widely used in the English language but have little contribution to the language meaning, e. g. "and, the, a, that, in, of". They can be a distraction to the language models, so they are pruned in many NLP tasks and information retrieval (search engine).
– **Special Character Removal**. Special characters are non-alphanumeric characters that could be noise to the language processing models. A strict policy to remove special characters is to keep only the alphanumeric characters in the text.
– **Digits Removal**. Depending on the tasks, sometimes digits are also considered as special characters that need to be removed.

Table 5: **Proceedings subset.** All papers accepted and published by the corresponding conferences, sorted alphabetically by venue.

Venue	Area	#Samples	Venue	Area	#Samples
CCS 2012	S	81	CCS 2013	S	105
CCS 2014	S	114	CCS 2015	S	128
CCS 2016	S	137	CCS 2017	S	151
CCS 2018	S	134	CCS 2019	S	149
CCS 2020	S	121	CCS 2021	S	196
CCS 2022	S	218	CCS 2023	S	234
CCS 2024	S	328	CCS 2025	S	316
Crypto 2018	S	79	Crypto 2019	S	81
CVPR 2013	V	471	CVPR 2014	V	539
CVPR 2015	V	597	CVPR 2016	V	643
CVPR 2017	V	783	CVPR 2018	V	978
CVPR 2019	V	1294	CVPR 2020	V	1466
EuroCrypt 2018	S	69	EuroCrypt 2019	S	76
FOCS 2018	T	86	HPCA 2019	A	37
ICCV 2013	V	454	ICCV 2015	V	525
ICCV 2017	V	621	ICCV 2019	V	1075
ICCV 2021	V	1612	MobiCom 2017	N	35
MobiCom 2018	N	42	NDSS 2013	S	47
NDSS 2014	S	55	NDSS 2015	S	50
NDSS 2016	S	60	NDSS 2017	S	68
NDSS 2018	S	71	NDSS 2019	S	89
NDSS 2020	S	88	NDSS 2021	S	90
NDSS 2022	S	83	NDSS 2023	S	94
NDSS 2024	S	140	NDSS 2025	S	211
PPoPP 2018	P	28	SIGCOMM 2015	N	43
SIGCOMM 2016	N	39	SIGCOMM 2017	N	38
SIGCOMM 2018	N	40	SIGKDD 2015	M	48
SP 2013	S	38	SP 2014	S	44
SP 2015	S	55	SP 2016	S	55
SP 2017	S	60	SP 2018	S	63
SP 2019	S	84	SP 2020	S	104
SP 2021	S	115	SP 2022	S	147
SP 2023	S	194	SP 2024	S	261
USENIX Sec 2013	S	44	USENIX Sec 2014	S	67
USENIX Sec 2015	S	67	USENIX Sec 2016	S	71
USENIX Sec 2017	S	85	USENIX Sec 2018	S	100
USENIX Sec 2019	S	112	USENIX Sec 2020	S	157
USENIX Sec 2021	S	246	USENIX Sec 2022	S	256
USENIX Sec 2023	S	422	USENIX Sec 2024	S	417
USENIX Sec 2025	S	407			
Total					**19028**

Legend: Areas: (A) Computer Architecture, (M) Data Mining; (N) Networking; (P) Parallel Programming; (S) Security; (T) Theory; (V) Computer Vision.