No more Reviewer #2: Subverting Automatic Paper-Reviewer Assignment using Adversarial Learning

Thorsten Eisenhofer*[†], Erwin Quiring*[†‡], Jonas Möller[§], Doreen Riepel[†],
Thorsten Holz[¶], Konrad Rieck[§]

[†] *Ruhr University Bochum*
[‡] *International Computer Science Institute (ICSI) Berkeley*
[§] *Technische Universität Berlin*
[¶] *CISPA Helmholtz Center for Information Security*

Abstract

The number of papers submitted to academic conferences is steadily rising in many scientific disciplines. To handle this growth, systems for automatic *paper-reviewer assignments* are increasingly used during the reviewing process. These systems use statistical topic models to characterize the content of submissions and automate the assignment to reviewers. In this paper, we show that this automation can be manipulated using adversarial learning. We propose an attack that adapts a given paper so that it misleads the assignment and selects its own reviewers. Our attack is based on a novel optimization strategy that alternates between the feature space and problem space to realize unobtrusive changes to the paper. To evaluate the feasibility of our attack, we simulate the paper-reviewer assignment of an actual security conference (IEEE S&P) with 165 reviewers on the program committee. Our results show that we can successfully select and remove reviewers without access to the assignment system. Moreover, we demonstrate that the manipulated papers remain plausible and are often indistinguishable from benign submissions.

1 Introduction

Peer review is a major pillar of academic research and the scientific publication process. Despite its well-known weaknesses, it is still an essential instrument for ensuring high-quality standards through the independent evaluation of scientific findings [6, 27, 51]. For this evaluation, a *submission* is assigned to a group of reviewers, taking into account their expertise, preferences, and potential biases. For conferences, this assignment is traditionally carried out by a program chair, while for journals, the task is performed by an editor. This mechanism has proven effective in the past, but is becoming increasingly difficult to realize as research communities grow. For example, the number of papers submitted to top-tier security conferences is increasing exponentially, reaching over 3,000 submissions in 2020. Likewise, the number of reviewers continuously grows for all major security conferences [5].

To handle this growth, conference management tools have become indispensable in peer review. They allow reviewers to bid for submissions and support the program chair to find a good assignment based on a best-effort matching. Unfortunately, even these tools reach their limit when the number of submissions continues to grow and manual bidding becomes intractable, as for example, in the area of machine learning. Major conferences in this area regularly have over 10,000 submissions that need to be distributed among more than 7,000 reviewers [31]. For this reason, conference management tools are increasingly extended with automatic systems for *paper-reviewer assignment* [14, 44]. These systems use topic models from machine learning to assess reviewer expertise, filter submissions, and automate the assignment process.

In this work, we show that this automation can be exploited to manipulate the assignment of reviewers. In contrast to prior work that focused on bid manipulations and reviewer collusion [24, 57], our attack rests on adversarial learning. In particular, we propose an attack that adapts a given paper so that it misleads the underlying topic model. This enables us to reject and select specific reviewers from the program committee. To reach this goal, we introduce a novel optimization strategy that alternates between the feature space and problem space when adapting a paper. This optimization allows us to preserve the semantics and plausibility of the document, while carefully changing the assignment of reviewers.

Our attack consists of two alternating steps: First, we aim at misleading the topic model employed in current assignment systems [14, 44]. This model defines a latent topic space that is difficult to attack because neither gradients nor an explicit decision boundary exist. To address this problem, we develop a search algorithm for exploring the latent space and manipulating decisions in it. As a counterpart, we introduce a framework for modifying papers in the problem space. This framework provides several transformations for adapting the paper's content, ranging from invisible comments to synonym replacement and generated text. These transformations enable us to preserve the paper's semantics, while gradually changing the assignment of reviewers.

[*] Shared first authorship

To empirically evaluate the practical feasibility of our attack, we simulate the paper-reviewer assignment of the 43rd IEEE Symposium on Security and Privacy (IEEE S&P) with the original program committee of 165 reviewers in both a *black-box* and a *white-box* threat scenario. As the basis for our attacks, we consider 32 original submissions that are publicly available with LaTeX source code.

Our white-box adversary achieves an alarming performance: we can successfully remove *any* of the initially assigned reviewers from a submission, and even scale the attack to completely choose *all* reviewers in the automated assignment process. In the black-box scenario, we can craft adversarial papers that transfer to an unknown target system by only using public knowledge about a conference. We achieve a success rate of up to 90% to select a reviewer and 81% to reject one. Furthermore, we demonstrate that the attack remains robust against variations in the training data.

Our work points to a serious problem in the current peer review process: With the application of machine learning, the process inherits vulnerabilities and becomes susceptible to new forms of manipulation. We discuss potential defenses: (1) For the feature space, robust topic modeling may limit the attacker's capabilities and (2) for the problem space, we recommend using optical character recognition (OCR) techniques to retrieve the displayed text. Nevertheless, these safeguards cannot completely fend off our manipulations and reviewers should be made aware of this threat.

Contributions. We make the following key contributions:

- *Attack against topic models.* We introduce a novel attack against topic models suitable for manipulating the ranking of reviewers. The attack does not depend on the availability of gradients and explores the latent topic space through an efficient beam search.

- *Problem-space transformations.* Our attack ensures that both the semantics and plausibility of the generated adversarial papers are preserved. This goal is achieved by a variety of transformations that carefully manipulate the document format and text of a submission.

- *Adversarial papers.* We present a method for constructing adversarial papers in a black-box and white-box scenario, unveiling a serious problem in automatic reviewer assignment. The attack rests on a novel hybrid approach to construct adversarial examples in discrete domains

Examples of the created adversarial papers are provided at https://github.com/rub-syssec/adversarial-papers. We also make our code and artifacts publicly available here.

2 Technical Background

Let us start by reviewing the necessary background for the design of our attack, covering the process of paper-review assignment and the underlying topic modeling.

Systems for paper-reviewer assignment. To cope with the abundance of submissions, several concepts have been proposed to assign reviewers to submissions [e.g., 30, 34, 35, 52]. In practice, the most widely used concept is *The Toronto Paper Matching System* (TPMS) by Charlin and Zemel [14]. Because of its high-quality assignments and direct integration with Microsoft's conference management tool CMT [15], TPMS is used by numerous conferences in different fields, including ACM CCS in 2017–2019 and NeurIPS/ICML. TPMS can be considered the de facto standard for automatic matching of papers to reviewers. Unfortunately, the implementation of TPMS is not publicly available and thus we focus in this work on *Autobid* [44], an open-source realization of the TPMS concept. Autobid closely follows the process described by Charlin and Zemel [14]. The system has been designed to work alongside HotCRP [26] and was used to support reviewer assignment at the IEEE Symposium on Security and Privacy (S&P) in 2017 and 2018.

Technically, TPMS and Autobid implement a processing pipeline similar to most matching concepts: (a) the text from the submission document is extracted and cleansed using natural language processing, (b) the preprocessed text is then mapped to the latent space of a topic model, and finally (c) an assignment is determined by deriving a ranking of reviewers. In the following, we review these steps in detail.

(a) Text preprocessing. When working with natural languages, multiple steps are required to bring text into a form suitable for machine learning (see Figure 1). As paper submissions can be provided in different formats, the pipeline starts by extracting text from the underlying document, typically the PDF format. This original document resides in the problem space of our attack and is denoted as z in the following. Autobid employs the tool `pdftotext` for this task, which is used in our evaluation in Section 4. The extracted text is then normalized using a preprocessor function ρ. Typically, it is tokenized, converted to lowercase, and stemmed [36]. Subsequently, stop words are removed so that each submission is now represented as a sequence of filtered stems. Autobid employs the NLTK package [8] to perform this task.

Finally, a feature extractor Φ maps the input $\rho(z)$ to a bag-of-words vector $\mathbf{x} \in \mathbb{N}^{|\mathcal{V}|}$ with $\mathcal{V}$ being the vocabulary formed over all words (stems). That is, a submission is represented by a high-dimensional vector whose individual dimensions reflect the count of words. Although this representation is

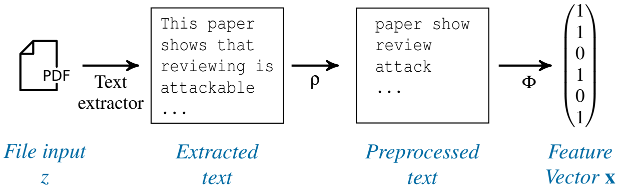


Figure 1: Text preprocessing in paper-reviewer assignment.

frequently applied in supervised learning, the high dimensionality is problematic for unsupervised learning and complicates determining topics in the submission.

(b) Topic modeling. The key to matching reviewers to papers is the automatic identification of topics in the text. This unsupervised learning task is denoted as *topic modeling*. While there exist several algorithms for this modeling, many assignment systems, including TPMS and Autobid, use *Latent Dirichlet Allocation* (LDA). LDA is a Bayesian probabilistic method for topic modeling that allows representing a document as a low-dimension mixture of latent topics. Formally, we define this representation as a function

$$\Gamma\colon \mathbb{N}^{|\mathcal{V}|} \longrightarrow \mathcal{T}, \quad \mathbf{x} \mapsto \theta_{\mathbf{x}}$$

mapping a bag-of-words vector $\mathbf{x}$ to a low-dimensional vector space $\mathcal{T}$, whose dimensions reflect different topics.

Generally, LDA is modeled as a generative probabilistic process [9, 16, 22]. It assumes a corpus D of documents and models each document as a random mixture over a set of latent topics T. A topic t is characterized by a multinomial distribution over the vocabulary $\mathcal{V}$, and drawn from a Dirichlet distribution $\phi_t \sim \mathit{Dirichlet}(\beta)$ with the prior β. The Dirichlet prior is usually sparse (i.e., $\beta < 1$) to model that words are not uniformly assigned to topics. Given these topics, for each document $\mathbf{x} \in D$, a distribution of topics $\theta_{\mathbf{x}} \sim \mathit{Dirichlet}(\alpha)$ is drawn. Again, the prior α is sparse to account for that documents are usually only associated with a small number of topics. Finally, for each word $w_i \in \mathbf{x}$, a topic $t_i \sim \mathit{Multinom}(\theta_{\mathbf{x}})$ is selected and the observed word $w_i \sim \mathit{Multinom}(\phi_{t_i})$ is drawn. This process can be summarized by the joint probability

$$P(\mathbf{w}, \mathbf{t}, \theta, \phi | \alpha, \beta) = P(\phi|\beta) P(\theta|\alpha) P(\mathbf{t}|\theta) P(\mathbf{w}|\phi_{\mathbf{t}}) \quad (1)$$

with $\mathbf{w} = (w_1, \ldots, w_{|\mathbf{x}|})$ and $\mathbf{t} = (t_1, \ldots, t_{|\mathbf{x}|})$.

To create a topic model in practice, we need to reverse this process and learn the posterior distribution of the latent variables $\mathbf{t}$, θ, and ϕ given the *observed* documents D. Specifically, we need to solve

$$P(\theta, \phi, \mathbf{t} | \mathbf{w}, \alpha, \beta) = \frac{P(\theta, \phi, \mathbf{t}, \mathbf{w} | \alpha, \beta)}{P(\mathbf{w}|\alpha, \beta)}. \quad (2)$$

Solving this equation is intractable as the term $P(\mathbf{w}|\alpha, \beta)$ cannot be computed exactly [9]. To address this, different approximated techniques, such as variational inference [9, 22] or Gibbs Sampling [16], are typically used for implementations of LDA. Autobid builds on variational inference based on the implementation of GenSim [48].

For the feature vector $\mathbf{x}$ of a new submission, the same technique—conditioned on the corpus D—is used to compute the corresponding topic mixture $\theta_{\mathbf{x}}$. Attacking this process is challenging, as no gradients or other guides for moving in the direction of particular topics are directly available. Hence, we develop a new search algorithm for subverting the topic assignment of LDA in Section 3.1.

(c) Paper-reviewer assignment. Finally, the topic model is used to estimate the reviewer expertise and automate the matching of submissions to reviewers. More specifically, let $\mathcal{R}$ be the set of all potential reviewers and $\mathcal{S}$ a set of submissions $\mathbf{x} \in \mathbb{N}^{|\mathcal{V}|}$. For each reviewer r, we collect an archive $A_r \in \mathbb{N}^{|\mathcal{V}|}$ representative of the reviewer's expertise and interests. Since researchers are naturally described best by their works, this could, for example, be a selected set of previously published papers. The corresponding archives are modeled as a union over all papers.

For each pair of reviewer r and submission $\mathbf{x}$, a *bidding score* $b_{r,\mathbf{x}}$ is calculated. This score reflects the similarity between the reviewer's archive A_r and a submission $\mathbf{x}$: the more similar, the higher the score. Given the topic extractor $\Gamma(\cdot)$, a reviewer r and a submission $\mathbf{x}$, Autobid defines the bidding score as the following dot-product

$$b_{r,\mathbf{x}} := \Gamma(A_r) \cdot \Gamma(\mathbf{x})^\top. \quad (3)$$

Subsequently, these bidding scores are used for the final assignment $\mathsf{A} \in \{0,1\}^{|\mathcal{R}| \times |\mathcal{S}|}$ with the goal to maximize the similarity between reviewers and submissions. In this phase, additional constraints are included: the assignment is subjected to (1) the targeted number of reviewers $\mathsf{L}_{\mathbf{x}}$ assigned to a submission and (2) the maximum number of submissions L_r assigned to a reviewer. More formally, we can describe the assignment as the following bipartite matching problem:

$$\begin{aligned}
\underset{\mathsf{A}}{\text{maximize}} \quad & \sum_r \sum_{\mathbf{x}} b_{r,\mathbf{x}} \cdot \mathsf{A}_{r,\mathbf{x}} \\
\text{subject to} \quad & \mathsf{A}_{r,\mathbf{x}} \in \{0,1\} \;\forall r, \mathbf{x} \\
& \sum_r \mathsf{A}_{r,\mathbf{x}} \le \mathsf{L}_{\mathbf{x}} \;\forall \mathbf{x} \\
& \sum_{\mathbf{x}} \mathsf{A}_{r,\mathbf{x}} \le \mathsf{L}_r \;\forall r
\end{aligned}$$

This optimization problem can then be reformulated and efficiently solved with *Linear Programming (LP)* [54].

3 Adversarial Papers

We proceed to introduce our approach for subverting the paper-reviewer assignments. To this end, we first define a threat model for our attack, and then examine challenges and required steps to control the matching.

Threat model. We consider a scenario where the adversary only modifies her submission—the *adversarial paper*—to manipulate the assigned reviewers. We assume two representative classes of adversaries with varying degrees of knowledge. First, we focus on *white-box adversaries* with complete access to the assignment system, including the trained model and reviewer archives. This represents a very strong class of adversaries and allows us to generally study the strength as well as limitations of our attack against assignment systems.

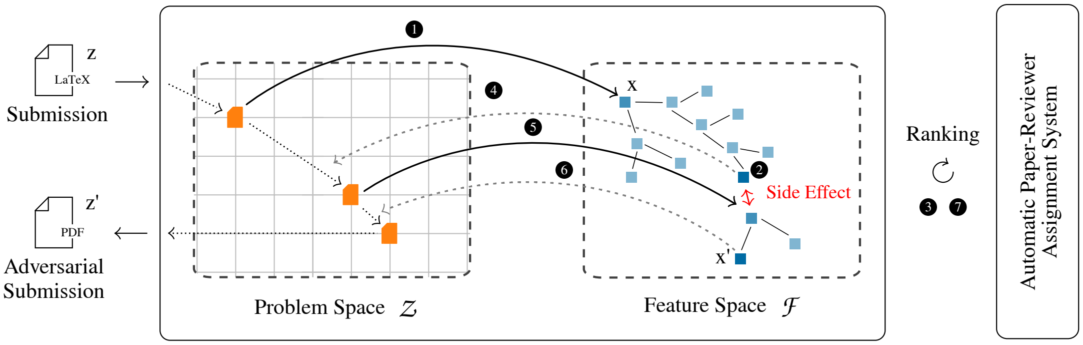

z
LaTeX
Submission
z'
PDF
Adversarial Submission
❶
❹
❺
❻
x
❷
Side Effect
x'
Problem Space $\mathcal{Z}$
Feature Space $\mathcal{F}$
Ranking
❸ ❼
Automatic Paper-Reviewer Assignment System

Figure 2: **Feature-problem-space attack.** For a submission z, we construct an adversarial submission z' that leads to a targeted assignment of reviewers. Our attack alternately switches between $\mathcal{Z}$ and $\mathcal{F}$. In step ❶, we extract word counts $\mathbf{x}$ from submission z, and use a search algorithm to change $\mathbf{x}$ in $\mathcal{F}$ to obtain the desired ranking (step ❷). To guide this search, we query the paper-reviewer assignment system for scores (step ❸). Next, we realize the modifications in the problem space $\mathcal{Z}$ and manipulate z (step ❹). Projecting the resulting submission back to $\mathcal{F}$, the submission vector will be shifted due to side effects and transformation limitations (step ❺). This shift is considered by continuing the search process from this new position and repeating this process iteratively (step ❻) until we obtain a valid adversarial submission z' (step ❼).

Second, we study the more realistic scenario with a *black-box adversary*. The adversary is assumed to have only a general knowledge about the assignment system (i. e., AutoBid is an open-source project [44]). No access to the training data and learned model is given. In this setting, we envision adversaries that exploit public information about a conference, such as knowledge about the program committee.

Challenges. The adversary has to operate both in the problem space $\mathcal{Z}$ and the feature space $\mathcal{F}$. The former consists of the input objects (e. g., the LaTeX source files of the paper); the latter contains the feature vectors that are used as inputs for the learning system. In contrast to domains like image recognition, the mapping from the problem space to the feature space is not bijective, i. e., there is no one-to-one correspondence between $\mathcal{Z}$ and $\mathcal{F}$. This poses a challenge for the adversary because a promising feature vector may not be mapped to a valid submission. A further obstacle is that some modifications in the problem space cannot be applied without side effects: If an adversary, for instance, adds a sentence to include a particular word, she inevitably adds other words that change the feature vector.

To deal with these challenges, we introduce a *hybrid* optimization strategy that alternates between the feature-space and problem-space representations of the attack submission. This optimization enables us to preserve the semantics and plausibility of the document, while at the same time gradually changing the assignment of reviewers. A general overview of our attack is outlined in Figure 2. Second, we transfer problem-space restrictions to the feature space. In this way, we resolve restrictions in a generic manner without adjusting our problem-space transformations.

Attack goals. Given a submission z, our goal is to find an adversarial paper z' that leads to the adversary's targeted review assignment. In the feature space, we thus want to manipulate the set of assigned reviewers $R_{\mathbf{x}}$. That is, we want to *select* and *reject* arbitrary reviewers to be included in respectively excluded from $R_{\mathbf{x}}$. Formally, we define two sets R_{sel} and R_{rej} and our goal is to find a vector $\delta \in \mathcal{F}$ such that the modified word counts $\mathbf{x}' := \mathbf{x} + \delta$ fulfill

$$\begin{aligned} r \in R_{\text{sel}} &\Rightarrow r \in R_{\mathbf{x}'}, \text{ and} \\ r \in R_{\text{rej}} &\Rightarrow r \notin R_{\mathbf{x}'}, \forall r \in \mathcal{R}. \end{aligned} \tag{4}$$

We require every reviewer $r \in R_{\text{sel}}$ is included in $R_{\mathbf{x}'}$ and likewise every reviewer $r \in R_{\text{rej}}$ excluded from $R_{\mathbf{x}'}$. In addition, we take care that the targeted solution is feasible with $|R_{\text{sel}}| \leq \mathsf{L}_{\mathbf{x}}$ and $|R_{\text{rej}}| \leq |\mathcal{R}| - \mathsf{L}_{\mathbf{x}}$.

Furthermore, we restrict the modifications to $||\delta||_1 \leq L_1^{\max}$ and $||\delta||_\infty \leq L_\infty^{\max}$. The L_1 constraint limits the amount of modifications to the submissions and makes the attack less suspicious. Similarly, the L_∞ constraint restricts the maximum change to a single feature, so that a word is not included too frequently. Finally, with respect to the concrete assignment process, we assume an automatic matching that always selects the reviewers with the highest assignment scores. We note that this assumption can be relaxed, as shown by Jecmen et al. [24], and combined with colluding reviewers. We further discuss the impact of concurring submissions in Section 5.

For manipulations in the problem space, we design various transformations for adapting the submission z. We denote a single transformation by $\omega : \mathcal{Z} \longrightarrow \mathcal{Z},\ z \mapsto z'$, where multiple transformations can be chained together as $\Omega = \omega_k \circ \cdots \circ \omega_2 \circ \omega_1$. To avoid transformations from creating artifacts and visible clues, we introduce the following problem-space constraints: First, we need to *preserve the semantics* of the text, so that the paper is still a meaningful submission. Second, we add a *plausibility* constraint, that is, the modifications should be as inconspicuous as possible. We summarize the constraints as Υ and write $\Omega(z) \models \Upsilon$ if a transformation sequence Ω on a submission fulfills these constraints.

Optimization problem. We arrive at the following optimization problem for generating adversarial examples, integrating constraints from the problem space and the feature space:

$$\text{subject to} \quad \begin{aligned} & r \in R_{\text{sel}} \Rightarrow r \in R_{\mathbf{x}'}, \text{ and} \\ & r \in R_{\text{rej}} \Rightarrow r \notin R_{\mathbf{x}'}, \forall r \in \mathcal{R} \\ & ||\delta||_1 \leq L_1^{\max} \text{ and } ||\delta||_\infty \leq L_\infty^{\max} \\ & \Omega(z) \models \Upsilon \end{aligned} \tag{5}$$

with $\mathbf{x} = \Phi \circ \rho(z)$, $\mathbf{x}' = \Phi \circ \rho(\Omega(z))$, and $\delta = (\mathbf{x}' - \mathbf{x})$. We proceed to realize this optimization strategy by first introducing our attack in the feature space and then in the problem space, before merging both components.

3.1 Feature Space

In an automatic paper-reviewer assignment system, the set of reviewers $R_\mathbf{x}$ for a submission is determined by the computed assignment scores $b_{r,\mathbf{x}}$ between reviewers r (characterized by their archives A_r) and the submission vector $\mathbf{x}$:

$$b_{r,\mathbf{x}} := \Gamma(A_r) \cdot \Gamma(\mathbf{x})^\top \tag{6}$$

To change the assignment score and thus affect the matching, we can only influence the extracted high-level features $\Gamma(\mathbf{x})$ since A_r is fixed for a given set of $\mathcal{R}$. However, even when we have full control over $\Gamma(\mathbf{x})$, changing the relative ordering—the *ranking*—between reviewers is not straightforward. For instance, suppose we have two reviewers r_1 and r_2 that share most topics (i.e., $\Gamma(A_{r_1}) \approx \Gamma(A_{r_2})$), adjusting $\Gamma(\mathbf{x})$ in this case will have a similar effect on both. In particular, if we naïvely try to increase the assignment score from r_1, we simultaneously also increase the score of r_2 and vice versa. Even if reviewers are not working in the same area, their topic distributions often partially overlap, as their research builds on similar principles and concepts. Hence, to modify the ranking we need to carefully maneuver the submission in the feature space. This is significantly more challenging compared to attacking a classification, as we need to both attack the model's prediction while simultaneously considering effects on concurring reviewers.

Our attack is further complicated by the fact that altering the topic distribution itself is a challenging task, since we need to make changes in the latent topic space. For LDA, this distribution $\Gamma(\mathbf{x}) = \theta_\mathbf{x}$ is computed using a probabilistic inference procedure. Thus, typical gradient-style attacks are not applicable. Indeed, Zhou et al. [59] even show that the manipulation of this inference is *NP-hard*. Moreover, LDA typically assigns only a small weight to individual words, so an attacker is required to manipulate a comparatively large set of words for subverting the topic assignment.

To address both of these challenges, we use a stochastic beam search. For a given submission vector $\mathbf{x}$, we start with an empty modification vector δ which is extended in each iteration until we find a successful submission vector $\mathbf{x}' := \mathbf{x} + \delta$ or a maximum number of iteration I is reached. During this search, we consider B candidate vectors in parallel and select successors after each iteration with a probability increasing as a function of the candidates' loss.

Loss. For our search, we define the following loss function to evaluate the quality of a submission $\mathbf{x}$ in terms of the objective from Equation 4 that incorporates the selection and rejection of reviewers:

$$\ell := \ell_{\text{sel}} + \ell_{\text{rej}} \tag{7}$$

For selected reviewers, the loss ℓ_{sel} is reduced when the assignment scores $b_{\hat{r},\mathbf{x}}$ increase or when the ranks of the reviewers improve (i. e., when reviewers ascend in the ranking):

$$\ell_{\text{sel}} := \sum_{\hat{r} \in R_{\text{sel}}} \text{rank}_\mathbf{x}^{\hat{r}} \cdot (1 - b_{\hat{r},\mathbf{x}}) \tag{8}$$

where $\text{rank}_\mathbf{x}^{\hat{r}}$ is the rank of reviewer $\hat{r}$ for submission $\mathbf{x}$. Similarly, for rejected reviewers the loss ℓ_{rej} is reduced when the assignment scores $b_{\check{r},\mathbf{x}}$ decrease:

$$\ell_{\text{rej}} := \sum_{\check{r} \in R_{\text{rej}}} \max(\text{rank}_\mathbf{x}^{\text{rej}} - \text{rank}_\mathbf{x}^{\check{r}}, 0) \cdot (b_{\check{r},\mathbf{x}} - b_{r_{\text{rej}},\mathbf{x}}) \tag{9}$$

where $\text{rank}_\mathbf{x}^{\text{rej}}$ is the target rank for a rejected reviewer (i. e., rejected reviewer are pushed down towards this rank) and $b_{r_{\text{rej}},\mathbf{x}}$ is the corresponding assignment score. This loss is designed to focus on reviewers that are far off, but simultaneously allows reviewers to provide "room" for following reviewers, for example, when we want to move a group of reviewers upwards/downwards in the ranking.

We consider a submission vector $\mathbf{x}$ successful when the objective from Equation 4 is fulfilled. At this point, we are naturally just at the boundary of the underlying decision function. To make the submission vector more *resilient*, we could continue to decrease the loss. However, since we already successfully ordered the reviewer (i. e., the ranking), we are more interested in maximizing the *margin* of selected and rejected reviewers to the border of $R_\mathbf{x}$. We denote this margin as γ and set $\ell := -\gamma$ whenever $\mathbf{x}$ satisfies Equation 4. Decreasing the loss is then equivalent to maximizing γ.

Candidate generation. A key operation of the beam search is the generation of new candidate vectors. We create a successor from a given submission by adding (respectively removing) k words to adjust topic distribution $\theta_\mathbf{x} = \Gamma(\mathbf{x})$ and ultimately the ranking of submission $\mathbf{x}$. To select words, we represent (broadly speaking) each reviewer by a set of *predictive* words and sample words that lie in the disjunction between a target and its concurring reviewers. An example of this is shown in Figure 3.

To construct these sets, we first represent each reviewer r by a *reviewer to words* distribution $\hat{\phi}_r$ over vocabulary $\mathcal{V}$. Intuitively, this distribution assigns each word the probability

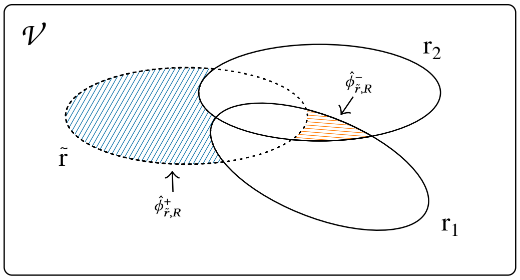


Figure 3: **Reviewer words.** Based on the topic model, we associate reviewer with a set of predictive words. Given target reviewer $\tilde{r}$ and a set of concurring reviewers $R = \{r_1, r_2\}$, we construct distributions $\hat{\phi}^+_{\tilde{r},R}$ and $\hat{\phi}^-_{\tilde{r},R}$. Sampling from these distributions yield words that are only predictive for reviewer $\tilde{r}$ respectively reviewers in R.

how predictive it is for r. Formally, we define the probability mass function for $\hat{\phi}_r$ as follows:

$$Q_r \colon \mathcal{V} \to \mathbb{R}, \quad w \mapsto \frac{\frac{1}{|T|}\sum_{t \in T} P(w \mid t)\; P(t \mid r)}{\sum_{w \in \mathcal{V}} \frac{1}{|T|}\sum_{t \in T} P(w \mid t)\; P(t \mid r)}$$

Remember that each topic t defines a distribution over $\mathcal{V}$ and each reviewer can be represented by $\Gamma(A_r)$. Q_r assigns each word the average probability over all topics T scaled by the relevance of topic t for reviewer r. Randomly sampling from $\hat{\phi}_r$ thus yield words with a probability given as a function of their *predictiveness* for r. In practice, $\mathcal{V}$ is typically large and most words are assigned with an insignificant probability. To improve performance, we therefore restrict $\hat{\phi}_r$ to the ν words with highest probability. We rescale the mass function to sum up to 1 so that $\hat{\phi}$ forms a valid distribution.

To select r, we could now simply add predictive words sampled from this distribution. However, as described earlier, naively doing this will likely have unwanted side effects because of concurring reviewers. To account for this, we further refine this distribution and simultaneously consider multiple reviewers. Let $\tilde{r}$ be a targeted reviewer and R a set of concurring reviewers. We want to restrict $\hat{\phi}_{\tilde{r}}$ to only include words that are predictive for $\tilde{r}$ but not for any reviewer in R. Specifically, we define $\hat{\phi}^+_{\tilde{r},R}$ with

$$Q^+_{\tilde{r},R} \colon \mathcal{V} \to \mathbb{R}, \quad w \mapsto \begin{cases} Q_{\tilde{r}}(w) & \text{if } Q_{\tilde{r}}(w) \neq 0 \land \\ & \quad \forall r \in R : Q_r(w) = 0 \\ 0 & \text{otherwise} \end{cases}$$

Subsequently, to form a valid probability mass function, we rescale $Q^+_{\tilde{r},R}$ to sum up to 1. Note for $R = \emptyset$ it follows $\hat{\phi}^+_{\tilde{r},R} = \hat{\phi}_{\tilde{r}}$. Sampling from $\hat{\phi}^+_{\tilde{r},R}$ only yields words that are predictive for $\tilde{r}$ but not R. Often we are also interested in the opposite case, i.e., words that are predictive for all reviewer in R but not for $\tilde{r}$ (e.g., when we want to remove words to promote $\tilde{r}$ in the ranking). Analogous, we define $\hat{\phi}^-_{\tilde{r},R}$ and write

$$Q^-_{\tilde{r},R} \colon \mathcal{V} \to \mathbb{R}, \quad w \mapsto \begin{cases} \frac{1}{|R|}\sum_{r \in R} Q_r(w) & \text{if } Q_{\tilde{r}}(w) = 0 \land \\ & \quad \forall r \in R : Q_r(w) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Again, we rescale $Q^-_{\tilde{r},R}$ to sum up to 1. For $R = \emptyset$, the distribution $\hat{\phi}^-_{\tilde{r},R}$ is not well defined, as its mass function always evaluates to 0 and we thus set $\hat{\phi}^-_{\tilde{r},R} := \hat{\phi}_{\tilde{r}}$. Figure 3 graphically depict this construction. For reviewer selection, we consider sets of concurring reviewer R that are close to r in the ranking. Specifically, we randomly sample M subsets from

$$R \subseteq Pow(\{r \mid \forall r \neq \tilde{r} \in \mathcal{R} : 0 \leq \text{rank}_{\tilde{r}} - \text{rank}_r - \upsilon \leq \omega\})$$

for a given reviewer window ω with offset υ. In other words, we exploit locality and focus on reviewer that are either before or close behind r in the ranking. For each subset, we create two candidates by (1) adding k words from $\hat{\phi}^+_{\tilde{r},R}$ respectively (2) remove k words from $\hat{\phi}^-_{\tilde{r},R}$. Reviewer rejection follows analogous with the distributions interchanged and sets sampled from

$$R \subseteq Pow(\{r \mid \forall r \neq \tilde{r} \in \mathcal{R} : -\omega \leq \text{rank}_{\tilde{r}} - \text{rank}_r + \upsilon \leq 0\})$$

Finally, for multiple target reviewer in R_{sel} and R_{rej}, we consider the union of candidates from individual reviewer.

3.2 Problem Space

The result of the feature space attack is a modification vector $\delta \in \mathcal{F}$ containing the words that have to be modified in the problem space. These words must be incorporated into an actual template PDF file $z' \in \mathcal{Z}$ such that both the semantics and plausibility constraints are satisfied. Fortunately, the assignment system obtains a document as input and not the raw text. This provides an adversary with more capabilities and flexibility. She can carefully manipulate the text of her submission as well as exploit weak spots in the text representation or document format.

Consequently, we divide the modifications into *text-level*, *encoding-level*, and *format-level* transformations—sorted according to their deniability. Text-level modifications operate on the actual text, so that only targeted modifications are possible. However, the modifications are deniable if the submission raises suspicion during reviewing. Encoding-level and format-level transformations manipulate the text representation and format, respectively, and enable large modifications, but are not deniable once detected. Table 1 lists the transformations implemented in our approach. For a detailed overview, we refer the reader to Appendix J.

Table 1: **Problem-space transformations.** Overview of transformations to realize modifications in the problem space. They are grouped by deniability (text, encoding, format) and the capability to add or delete words. For a detailed description, see Appendix J.

		Modification	
Type	Transformation	Add.	Del.
Text-level	Reference addition	●	○
	Synonym	●	●
	Spelling mistake	○	●
	Language model	●	○
Encoding-level	Homoglyph	○	●
Format-level	Hidden box	●	●

Text-level transformations. We begin with transformations that are based solely on changes to the visible text and applicable to any text format. As such, they cannot be readily recognized without a semantic analysis of the text.

(a) Reference addition. As the first transformation, we consider additions to the submission's bibliography. The transformation adds real references that contain the words to be added. As references, we use publications from security conferences and security-related technical reports. Our evaluation demonstrates that this transformation is very effective, while creating plausible and semantics-preserving changes to a paper. However, it introduces side effects, as not only selected words are added, but also parts of the conference names, authors, and titles. This motivates the hybrid search strategy that we outline in Section 3.3.

(b) Synonym. We develop a transformation that replaces a word with a *synonym*. To enhance the quality of the proposed synonyms, instead of using a general model for the English language [e. g. 25, 29, 49], we use a security-domain specific neural embedding that we compute on a collection of 11,770 security papers. Section 5 presents the dataset. This domain-specific model increases the quality of the synonyms, so that this transformation is also difficult to spot.

(c) Spelling mistake. As a third type of text-level manipulations, we implement a spelling-mistake transformation, which is common for misleading text classifiers [21, 33]. Here, we improve on prior work by trying to find typographical errors from a list of common misspellings [1] instead of introducing arbitrary mistakes. For example, the suffix *ance* is often confused with *ence*, so that “appearance” becomes the unobtrusive misspelling “appearence”. If we do not find such errors, we apply a common procedure from the adversarial learning literature: We either swap two adjacent letters or delete a letter in the word [21, 29, 33].

(d) Language model. Finally, we apply the large-scale unsupervised language model OPT [58] to create text containing the words to be added. To generate security-related text, we finetune the model using the corpus of 11,770 security papers. While the created sentences are not necessarily plausible, this transformation allows us to technically evaluate the possibility that an adversary creates new text to insert words. Given the increasing capabilities of language models, we expect the chances of creating plausible text to rise in the long run. Moreover, we assume that in practice attackers would manually polish the generated text to reduce their detection probability.

Encoding-level transformations. As the second class of transformations, we consider manipulations of the text encoding. These manipulations may include the substitution of characters, the application of unicode operations, or changes to the font face and color. For our implementation, we focus on *homoglyph* transformation, inspired by previous works that replaces characters with visually similar counterparts [19, 29]. By replacing a character with a homoglyph, we can remove selected words from the bag-of-words vector used for the topic model. Similarly, there are several other strategies for tampering with text encoding [10]. Since these manipulations also change only the visual appearance of the text, we consider homoglyphs as a representative example of the class of encoding-level transformations.

Format-level transformations. As the third class of transformations, we focus on changes specific to the underlying document format, such as accessibility features, scripting support, and parsing ambiguity [38]. As an example of this class of transformations, we consider *hidden boxes* in the PDF format. Our transformation relies on accessibility support with the latex package `accsupp` to define an invisible alternative text in a hidden box associated with a word. The text extractor processes the alternate text, while PDF readers display only the original word. This discrepancy allows an attacker to add words as alternate text. Likewise, she can put an empty alternative text over a word that should be removed.

Improved transformations. In addition, we exploit the preprocessing implemented by assignment systems. First, we benefit from stemming, so that the transformations only need to add or delete *stems* instead of words. This increases the possibilities to find suitable text manipulations. For example, an attacker can modify the words *attacker* or *attackable* to remove the feature *attack*, since both are reduced to the same stem. Second, we exploit the filtering of stop words. The hidden box transformation requires sacrificing a word for defining an alternative text. As stop words are not part of the feature vector, no side effects occur if they are changed.

3.3 Feature-Problem-Space Attack

We are now equipped with (i) a strategy to find modifications $\delta \in \mathcal{F}$ and (ii) transformations $\omega \in \mathcal{Z}$ to realize δ in a paper submission. The ultimately missing piece is an optimization strategy that brings these two components together. In general, this optimization is responsible for guiding the transformations towards the targeted assignment. In the following, we first present the basic principle of our applied strategy and then introduce two practical extensions.

Hybrid optimization strategy. Due to the challenges around the problem space and the feature space, we use a strategy that *switches alternately* between $\mathcal{Z}$ and $\mathcal{F}$. Figure 2 on page 4 schematically illustrates our alternating approach. For an initial submission z, the adversary extracts the features (step ❶) and performs a feature-space attack (step ❷ and ❸). As Φ is not invertible, the adversary then has to find suitable transformations in the problem space (step ❹) that realize the requested modifications. This leads to a new feature vector in $\mathcal{F}$ (step ❺). However, this vector is shifted due to side effects and limitations of the transformations. Consequently, the adversary continues her search from this new position and repeats the process iteratively until the target is reached or the maximum number of iterations have passed.

We note that side effects are not always negative as assumed by prior work [45]. In our evaluation, for example, we found that the additional words introduced by the reference transformation can further push a reviewer's rank towards the target, since the additional words may also relate to other selected reviewers, for example, due to co-authors or paper titles. However, the impact of side effects is difficult to predict in advance, so that an optimization strategy should be capable of dealing with positive as well as negative side effects.

Constraint mapping $\mathcal{Z} \rightarrow \mathcal{F}$. Our first extension to this hybrid strategy addresses the complexity of problem-space modifications. In practice, not every requested modification from $\mathcal{F}$ can be realized in $\mathcal{Z}$ with the implemented transformations due to PDF and LaTeX restrictions. For example, in LaTeX, homoglpyhs are not usable in the listing environment, while the hidden box is not applicable in captions or section titles. In general, such restrictions are difficult to predict given the large number of possible LaTeX packages. Instead of solving such shortcomings in the problem space by tediously adjusting the transformations to each special case, we resort to a more generic approach and transfer problem-space constraints back to the feature space. The transformers in $\mathcal{Z}$ first collect words that cannot be handled, which are then blocked from being sampled during candidate generation in $\mathcal{F}$.

Surrogate models. We introduce a second extension for the black-box scenario. In this scenario, the adversary has no access to the victim model. Still, she can leverage public information about the program committee, collect papers from its members, and assemble a dataset similar to the original training data. This allows her to train a *surrogate model* that enables preparing an adversarial paper without access to the assignment system. This strategy has been successfully used for attacks against neural networks [42]. However, in our case, this strategy is hindered by a problem: LDA models suffer from high variance [2, 37]. Even if the adversary had access to the original data, she would still get different models with varying predictions. This makes it unlikely that an adversarial paper computed for one model transfers to another.

As a remedy, we propose to use an ensemble of surrogate models to better approximate the space of possible LDA models. We run the attack simultaneously for multiple models until being successful against *all* surrogates. To this end, we extend the feature-space attack to multiple target models: (i) we create candidates for each surrogate model independently and consider the union over all surrogates and (ii) we compute the loss as the sum of individual losses over all surrogates. Intuitively, this increases the robustness of an adversarial paper and, consequently, improves the success rate that the attack transfers to the unknown victim model.

4 Evaluation

In the following, we evaluate the efficacy of the proposed approach to prepare adversarial papers. To this end, we simulate the automatic paper-reviewer assignment process of a real conference with the full program committee (PC). We consider two different scenarios: First, we demonstrate how a white-box attacker with full-knowledge about the target system can select and reject reviewers for a submission. Second, we consider a black-box adversary with only limited knowledge and no access to the trained assignment system. We show that such an adversary can generate effective surrogate models by exploiting public knowledge about the conference. Finally, we verify that the manipulated papers are plausible and preserve the semantics of the text.

Setup. We use Autobid [44] as an open-source realization of the TPMS concept [14]. We simulate an automatic assignment for the *43rd IEEE Symposium on Security and Privacy* with the full PC and set the paper load $\mathsf{L}_{\mathbf{x}} = 5$ (i. e., assign each submission to five reviewers). In contrast to the real conference, we assume a fully automated assignment without load balancing and conflicts (see Section 5). As we do not have access to the original submissions, we use the accepted papers as substitutes. In total, we find 32 papers on the arXiv e-Print archive with LaTeX source, which we use for our evaluation.

The PC consists of 165 persons. For each PC member, we construct an archive A_r of papers representative for the person's expertise by crawling their *Google Scholar* profile. We select 20 paper for each reviewer and compile the corpus as the union of these archives. To simulate a black-box scenario, we additionally generate *surrogate corpuses* that overlap with the original data between 0% and 100%. Appendix A describes this process in detail. In all cases, we train Autobid with the default configuration on a given corpus.

For each attack, we perform a grid search on its parameters to realize a reasonable trade-off between efficacy and efficiency. We start by relaxing any constraints on δ ($L_1^{\max} = \infty$ and $L_\infty^{\max} = \infty$) and run the attack with at most $S = 8$ transitions between the feature space and problem space (see Appendix B for details). All experiments are performed on a server with 256 GB RAM and two Intel Xeon Gold 5320 CPUs.

Performance measures. We use three measures to evaluate the attack's performance. First, we consider an adversarial paper z' to be *successful* if the constraints from Equation 5 are fulfilled. Second, to quantify modifications to the submission, we use two standard measures: L_1 and L_∞ norm. Given the modified word counts $\mathbf{x}' := \mathbf{x} + \delta$, these are computed as

$$||\delta||_1 = \sum_i |\delta_i| \text{ and } ||\delta||_\infty = \max_i |\delta_i| . \tag{10}$$

L_1 is the absolute number of modified words and provides a general overview on the total amount of modifications. Intuitively, we are interested in minimizing L_1 to make an attack less suspicious. Similarly, L_∞ is the maximum change in a single dimension (i.e., a single word) and ensures that a single word is not included too frequently. Third, we assess the *semantics* and *plausibility* of the manipulated papers in a user study with security researchers.

4.1 White-box Scenario

In our first scenario, we focus on a white-box scenario and consider three attack objectives: (1) selection, (2) rejection, and (3) substitution of a reviewer. For these objectives, we focus on reviewers that are already "close" to a submission in the assignment system. For example, a paper on binary analysis would raise suspicion if it would get assigned to a reviewer with a cryptography background.

We use the initial assignment scores of the submission as a proxy to simulate this setting. We determine potential reviewers by computing the ranking for the unmodified submission and consider the 10 reviewers with highest scores. To study objective (1), we sample reviewers from the ranks 6–10 and attempt to get them assigned to the submission. Analogously, for objective (2), we select reviewers from the ranks 1–5 and aim at eliminating their assignment. Finally, for objective (3), we first select a reviewer for removal and then a counterpart for selection. We repeat this procedure 100 times with random combinations of papers and reviewers for each objective. Moreover, to account for the high variance of LDA, we train the topic model 8 times and average results in the following.

Table 2: **Feature-space search.** We compare our attack with two baselines: hill climbing and morphing. For this comparison, we consider three attack objectives: (1) selecting, (2) rejecting, and (3) substituting of reviewers.

	Selection		Rejection		Substitution	
L_1 norm						
Our attack	704	×1.00	1032	×1.00	2059	×1.00
Hill climbing	1652	×2.35	2255	×2.18	5526	×2.68
Morphing	3059	×4.35	-	× -	-	× -
L_∞ norm						
Our attack	17	×1.00	43	×1.00	62	×1.00
Hill climbing	38	×2.22	44	×1.02	98	×1.58
Morphing	45	×2.63	-	× -	-	× -

Feature-space search. We start our evaluation by examining the feature-space search of our attack in detail. For this experiment, we consider format-level transformations that can realize arbitrary changes. Other transformations are evaluated later when we investigate the problem-space side of our attack.

The results of this experiment are presented in Table 2 and further detailed in Appendix C. We observe that our approach is very effective: 99.7 % of the attacks finish successfully with a median run-time of 7 minutes. The number of performed changes shows high variance, ranging between 9 and 22,621 adapted words. Despite this broad range, however, the average manipulation involves only between 704 and 1,032 words for objectives (1) and (2), respectively. For reference, an unmodified submission contains 7,649 words on average, so that the necessary changes for preparing an adversarial paper amount to 9% and 13% of the words.

Among the three objectives, we see a trend that selecting a reviewer is more efficient than rejecting one. Rejected reviewers have—per construction—a high assignment score, and hence share many topics with nearby reviewers. In contrast, for selected reviewers it is easier to determine topics with less side effects. The third scenario, where we both reject and select a reviewer, is naturally the hardest case. Generally, we observe that topic models based on LDA are comparatively robust against adversarial noise, in relation to neural networks which can be deceived into a misclassification by changing only a few words [e.g., 21, 29].

Baseline experiments. To put these numbers into perspective, we examine two baselines. First, we implement a hill climbing approach that directly manipulates the topic vector of a submission (cf. Equation 6) by sampling words from the topic-word distributions associated with a reviewer. For the second baseline, we consider an approach that morphs a target submission with papers that already contains the correct topic-word distribution. To find these papers, we compute all assignments of the training corpus and identify submissions to which the target reviewer is assigned. We then repeatedly select words from these submissions and expand our adversarial paper until we reach the target. In rare cases, we could not find papers in which the reviewer is highly rated. We exclude such cases from our experiments.

Considering all three objectives, the hill climbing approach shows a lower success rate: Only 92.2 % of the papers are successfully manipulated. The failed submissions either reach the maximum number of 1,000 iterations or get stuck in a local minimum. In successful cases, the attacker needs to introduce more than twice as many changes compared to our attack and the median L_1 norm increases from 704–2,059 to 1,652–5,526 words. For the morphing baseline, the attack is successful in only 91.1 % of the cases and again needs to introduce significantly more words. We find that the median L_1 norm increases by a factor of 4.35 with a maximum of 29,291 modified words for a single attack.

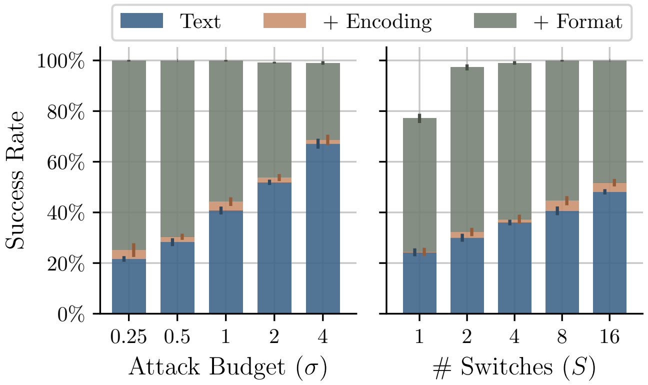


Figure 4: **Feature-problem-space attack.** We simulate the attack with differently scaled attack budgets σ (left) and $S = 8$ switches. We repeat the experiment (right) with the base budget $\sigma = 1$ and vary S. For both cases, we randomly select 100 targets from all three objectives that require $\leq 1{,}000$ changes in $\mathcal{F}$. We report the mean success rate over 8 repetitions.

Generalization of attack. To investigate the generalization of our attack, we repeat this experiment for a second real conference. In particular, we simulate the assignment of the *29th USENIX Security Symposium* with 120 reviewers. We consider 24 original submissions and construct targets as before. Results of this experiment are presented in Appendix D. We observe a similar performance across all three objectives, indicating the general applicability of our attack.

Scaling of target reviewers. Next, we scale the attack to larger sets of target reviewers and consider different combinations for selecting, rejecting, and substituting reviewers. We allow an attacker to select up to five target reviewers, which is equivalent to replacing *all* of the initially assigned reviewers. Furthermore, we allow the rejection of up to two reviewers. We focus again on close reviewers and randomly select 100 sets of targets per combination.

The results are summarized in Appendix E. The attack remains effective and we can successfully craft adversarial papers in most of the cases. We observe a clear trend that with increasing numbers of target reviewers, we need to perform more changes to the submission. For example, to select all five reviewers, in the median we need to modify 5,968 words. This is expected: we have to move the submission in the topic space from the initially-assigned reviewers to the targeted ones. By adding more reviewers, we include more constraints which results in a significant amount of modifications.

All transformations. So far, we have focused on format-level transformations to realize manipulations. These transformations exploit intrinsics of the submission format, which effectively allows us to make arbitrary changes to a PDF file. An attacker likely has access to similar transformations in any practical setting. In fact, robust parsing of PDF files has been shown to be a hard problem [e.g., 13]. However, we believe it is important for an attacker to minimize any traces and consider different classes of transformations as introduced in Section 3.2.

(a) Attack budget. For this experiment, we introduce an attack budget to describe the maximum amount of allowed modifications for a given transformation. This budget trades off the ability of a transformation to introduce changes with their conspicuousness, since too many (visible) modifications will likely lead to a rejected submission. In particular, we assume a maximum of 25 real and 5 adaptive added BibTeX entries, at most 25 replacements of words with synonyms, no more than 20 spelling mistakes, and up to 10 requested words on average through a text from a language model. In Section 4.3, we validate these parameters and assess if the resulting adversarial papers are unobtrusive to human observers.

As a result of the attack budget, we cannot realize arbitrary modifications, since their total amount is restricted. To study this in more detail, we consider the success rate as a function of the attack budget scaled with a factor σ between 0.25 and 4. During the attack, we split the budget equally across 8 feature-problem-space transitions. We require that targets are feasible with this budget and randomly select 100 targets from the three attack objectives that require $\leq 1{,}000$ changes in $\mathcal{F}$. Finally, we consider three different configurations: (1) text-level transformations, (2) text-level and encoding-level transformations, and (3) text-level, encoding-level, and format-level transformations combined. We do not restrict the budget for format-level transformations as these transformations are generally not visible.

The results are shown on the left side of Figure 4. For text-level transformations and text-level & encoding-level transformations, we see an increase in the success rate when the attack budget grows. For the base budget ($\sigma = 1$), 40.75% of the adversarial papers can be prepared with text-level transformations only. That is, no changes in the format and encoding are necessary for manipulating the reviewer assignment. This can be further improved by increasing the budget, for instance, 67.13% of the papers become adversarial by scaling it to 4. For smaller budgets, however, we observe that there is often not enough capacity to realize the required modifications. Still, using format-level transformations improves the success rate to 100% in almost all cases. In rare case, we observe that the attack gets stuck in a local minima. Interestingly, this is more likely with larger budgets. In these cases, the attack makes bigger steps per iteration which introduces more side effects. From the perspective of an attacker, this can be resolved by either increasing the number of switches or reducing the budget.

(b) Problem-feature-space transitions. To better understand the influence of the alternating search on the success rate of our attack, we conduct an additional experiment. In particular, we simulate our attack for different numbers of transitions $S \in \{1, 2, 4, 8, 16\}$ between the problem space and

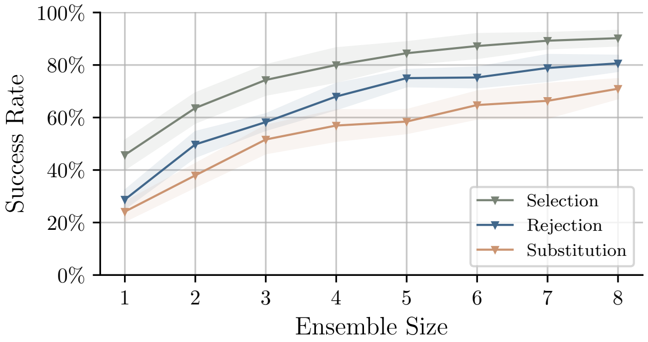


Figure 5: **Surrogate ensemble sizes.** We simulate the attack with varying numbers of surrogate models. For each ensemble size, we report the mean success rate over 8 target systems with 100 targets each for all three attack objective.

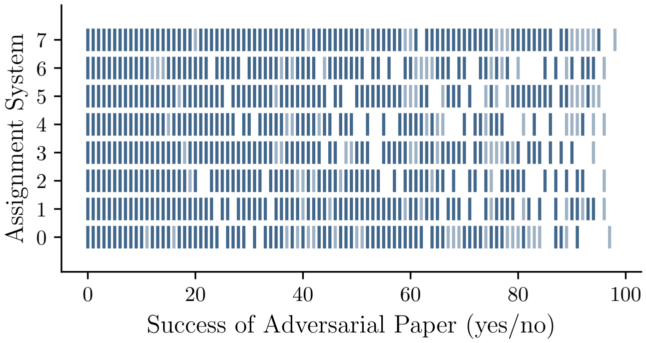


Figure 6: **Transferability.** We visualize the transferability of 100 adversarial paper among 8 target assignment systems. Attacks were performed with an ensemble size of 8 and we focus on the selection objective. Adversarial papers where the unmodified submission is already successful are displayed in light blue.

the feature space. We consider the same targets as before and set the attack budget to $\sigma = 1$.

The results of this experiment are depicted on the right side of Figure 4. Increasing the number of transitions has a significant effect on the success rate. For all configurations, we see a steady improvement when the number of problem-feature-space transitions increases. Notably, even the format-level transformations *require* multiple transitions in some cases. The success rate increases from 77.13%—with no transitions—to 100% when increasing S. By alternating between $\mathcal{F}$ and $\mathcal{Z}$ we share constraints between problem and feature space to find modifications that can be realized in the problem space. This further underlines that it is beneficial and in fact necessary to consider both spaces together.

4.2 Black-box Scenario

In practice, an attacker typically does not have unrestricted access to the target system. In the following, we therefore assume a black-box scenario and consider an adversary with only limited knowledge. In particular, this adversary cannot access the assignment system and its training data. Instead, we demonstrate that she could leverage her knowledge about the program committee and construct a surrogate dataset to train her own models for preparing adversarial papers.

The assignment systems AutoBid and TPMS do not specify how the corpus for training a topic model is constructed. They only require that the selected publications are representative of the reviewers. Hence, even if we do not know the exact composition of the training data, we can still collect a surrogate corpus of representative data with public information, such as recent papers of the PC members, and transfer our attack between models. In practice, the success of this transfer depends on two factors: (a) the stability of the surrogate models and (b) the overlap of publications between the original training data and the surrogate corpus.

Stability of surrogate models. The training of LDA introduces high variance [2, 37], so that adversarial papers naïvely computed against one model will likely not transfer to another. To account for this instability, we approximate the model space and consider an *ensemble* of surrogate models. That is, we run our attack simultaneously against multiple surrogate models trained on the same data. We focus on format-level transformations and repeat the attacks for all three objectives. We vary the number of models in the ensemble from 1 to 8 and consider an overlap of 70% between the underlying surrogate corpus and the original training data.

Figure 5 show the results of this experiment. Across all objectives, we see an improvement of the success rate when increasing the number of surrogate models. This is intuitive: the adversarial papers are optimized against all models and thus more likely to transfer to other models. This robustness, however, comes at a cost and the number of modifications increases as well. The median L_1 norm increases from 1,990 to 7,556 when considering 8 instead of a single surrogate model (see Appendix F).

As a result, an adversary in the black-box scenario must find a trade-off: If she needs a successful attack with high probability, she must sacrifice detectability and modify a large number of words. If, on the other end, she only wants to increase her chances for a specific assignment, she can operate without an ensemble and adapt only a few words.

To further study the transferability of our attack, we sample 100 target reviewer from the median ranking computed over 8 assignment systems and simulate the attack with an ensemble of 8 surrogates. Figure 6 visualizes how the resulting adversarial papers transfer among the target systems. 96% of the papers are successful against four or more target systems and 34 % are successful considering all 8 systems.

Overlap of surrogate corpus. To understand the role of the surrogate corpus, we finally repeat the previous experiment with varying levels of overlap. Surprisingly, the attack remains robust against variations in training data. The success rate only fluctuates slightly: 78.0% (100% overlap), 80.0% (70% overlap), 79.6% (30% overlap), and 82.8% (0% overlap). To explain this, we compute the cross-entropy of the reviewer-to-words distributions $\hat{\phi}_r$ for models trained on training data with different overlaps. We observe that the cross-entropy between models trained on the same dataset (i.e., 100% overlap) is in the same range compared to models trained on different data (cf. Appendix G for details). As LDA models trained on the same corpus already vary significantly, our attack seeks a robust solution that transfers well if the surrogate models have less overlap with the original training data.

4.3 Plausibility and Semantics

Finally, we empirically verify if the adversarial modifications are (a) plausible and (b) preserve the semantics of the text.

Study design. As dataset, we use the combined set of original and adversarial papers from our evaluation. In total, we select seven original papers and their adversarial counterparts, ensuring varying topics and transformations. The attack budget is $\sigma = 1.00$. Due to a limited number of participants, we focus on visible transformations (i.e. encoding-level and text-level) that a reviewer could detect. Each participant selects ("bids on") one paper. This selection cannot be changed afterwards and participants are secretly assigned either to the adversarial or to the unmodified version. Each participant will only check one paper to avoid potential bias and fatigue effects.

We design the review process along two phases. Our methodology here is inspired by the work from Bajwa et al. [4] and Sullivan et al. [53]. In the first phase, we request participants to write a mini-review (as a proxy task) for a given paper. In the second phase, we ask if they think the paper has been manipulated. Importantly, the answers of phase 1 cannot be changed. This two-phase separation allows us to observe two factors: First, we can analyze how suspicious adversarial papers are to an unaware reader. Second, once the reader is informed, we can learn about which transformations are noticeable and make our attack detectable. In each phase, we provide a template with questions on a numerical scale from 1–5, together with free text fields for justifying the rating. Participants are debriefed finally. We obtained approval from our institution's Institutional Review Board (IRB) and our study protocol was deemed to comply with all regulations.

Results. We recruited 21 security researchers (15× PhD students, 4× postdocs, 1× faculty, 1× other). All participants are familiar with the academic review process but have different review experience (7× have not written a review before, 4× between 1-2 reviews, 6× between 3-10, and 4× at least 10 reviews). The participants reviewed a total of 12 adversarial and 9 original submissions.

Figure 7 summarizes the results. Benign and adversarial submissions are rated similar across all review questions. No participant was certain that a paper was manipulated (i. e., gave it a ranking of 5) and only a single of the 12 manipulated submissions was flagged as suspicious with a rating of 4. This was justified with missing references and redundancy in the text—neither of which were introduced by our attack. Interestingly, this reviewer did notice the spelling mistake and language model transformer (when asked about the writing quality), but did not attribute this as a sign for manipulation. This is opposed to two false positive ratings of benign papers, which results in a overall detection precision of 33% with a recall of only 8%. This highlights the difficulty to detect any introduced modifications.

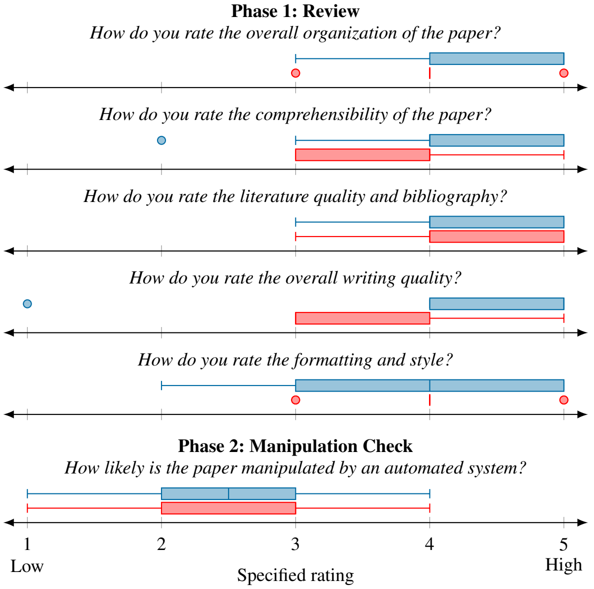


Figure 7: **Ratings of benign and adversarial papers.** For each question, the upper boxplot shows the ratings from the benign papers, the lower boxplot from the adversarial papers.

Finally, we check that the semantics of the papers are not changed. With the limited attack budget, only small, bounded changes are made to a paper. This is further supported by the organization and comprehensibility ratings in Figure 7, which are similar between manipulated and benign submissions.

5 Discussion

Our work reveals a notable vulnerability in systems for automatic paper-reviewer assignment. In the following, we discuss further aspects of our findings, including limitations, defenses, and the implications and benefits of an attack.

Committee size. We simulate an automatic assignment for two large security conferences with committees composed of 120 and 165 reviewers, respectively. Considering the current trend, it is likely that these conferences will continue to grow larger. In the following, we want to understand how an increased set of concurring reviewers impacts the attack. Therefore, we consider committees with 100–500 reviewers sampled from a pool of 528 reviewers (taken from USENIX and S&P committees between 2018–2022) with a total of 11,770 papers in the underlying corpus. Appendix H shows the number of required changes as a function of the committee size. Across the three objectives—selection, rejection, and substitution—we observe only a minor impact on the attack. The attack remains successful with a success rate between 98.00% – 98.92% and the number of required modifications remains largely unaffected. For the smallest committee considered (with 100 reviewers), we observe a slightly larger uncertainty. Intuitively, in this case the assignment is more dependent on the particular choice of the committee which gets averaged out for larger committees.

Load balancing and conflicts. Our attack focuses on the manipulation of assignment scores and assumes a direct matching from PC members to submissions. For a complete end-to-end attack, an attacker would also need to take load balancing of submissions and reviewer conflicts into account. For example, the target submission might compete with another paper on the same topic and get a different assignment despite a successful manipulation of the topic model.

Unfortunately, these obstacles are hard to model, as conflicts and the other submissions are typically not known to the adversary. Instead, we can generally improve the resilience of our attack. By increasing the margin γ of the target reviewer to others, we can make a matching assignment more likely. Interestingly, in this case, conflicts can be even seen as a simplification: if a target reviewer is the top candidate among all reviewers $\mathcal{R}$, she is also the top candidate for only a subset of reviewers (i.e., all unconflicted reviewers).

To further understand the role of this margin, we simulate the selection of a reviewer for different values of $\gamma \in \{0, 0.1, 0.2\}$ and varying numbers of concurring submissions between 200 and 1,000 (sampled from a hold-out corpus). We model the full assignment to maximize similarity subjected to load constraints as introduced in Section 2. We assume $\mathsf{L}_{\mathbf{x}} = 5$ reviews per paper and that each reviewer is assigned $\mathsf{L}_r = 10$ submissions. Appendix I shows the attack's success rate as a function of the number of concurring submissions. The attack remains effective but we observe a slight downward trend of its success rate. This is expected: with increasing number of submissions, there exist more similar paper that compete for a given reviewer. An attacker can account for this by (1) increasing the margin and, as the attack is undetectable (in general), an attacker could (2) further increase her chances by repeating the attack (e.g., resubmitting a rejected paper).

Paper corpus. We select *accepted* papers from IEEE S&P 2022 as basis for our evaluation. This selection leads to a potential bias, as rejected submissions are not considered. However, we do not expect any impact on our results. Papers follow a common structure, so that our transformations in LaTeX are applicable in general. The feature-space algorithm works on bag-of-words vectors, which is just another representation for any paper. In Appendix D, we test our attack with papers from the 29th USENIX Security Symposium and find no significant difference in our results.

Countermeasures and defenses. Our results show that systems based on topic models such as LDA have relatively strong robustness towards adversarial noise. This stands in stark contrast to neural networks, where changing only a few words can already lead to a misclassification [e.g., 21, 29]. However, our work also demonstrates that LDA-based systems are still vulnerable to adversarial examples and there is a need for appropriate defenses.

Unfortunately, text-level manipulations are challenging to fend off, as they can only be spotted on the semantic level. In our user study, participants often struggled to differentiate adversarial modifications from benign issues and an adversary can always *manually* rewrite an adversarial paper to further reduce the detection probability. Moreover, even completely machine generated text—such as done with our OPT-based transformer—is often indistinguishable from natural text [11, 40]. The underlying models are evolving rapidly and current state-of-the-art models such as InstructGPT [41] and Galactica [55] are now actively used for academic writing.

For encoding-level and format-level changes, however, defenses are feasible: The root cause of these manipulations is the disparity between human perception and parser-based text extraction. Thus, an effective defense needs to mimic a human reader as close as possible similar to defenses recently proposed for adversarial examples in other domains [e.g. 20]. To evaluate this defense, we replace the parser-based text extraction (`pdftotext`) with an optical character recognition (OCR) (`tesseract`). We observe that for the modified system the encoding-level and format-level attacks now completely fail, while the performance of the text-level attacks remains unaffected. At the same time, however, we observe a large increase in runtime. Compared to the parser-based extraction, OCR is orders of magnitude slower and needs an average time of 56 s for a single submission compared to 0.14 s with conventional text extraction.

Other countermeasures can be more tailored to the individual transformations: Flag usage of unusual font encodings to prevent homoglyph attacks, remove comment boxes and non-typeset pages in a preprocessing step, or automatically verify the bibliography entries using online bibliography databases.

Benefits and implications. Manipulating a submission comes with a considerable risk if the attack is detected. This can range from a desk reject over a submission ban at a specific venue to permanent damage of the authors' scientific reputation [50]. Nevertheless, recent incidents show that academic misconduct happens. Dishonest authors, for example, leveraged synthetic texts to increase the paper output [12]. Moreover, *collusion rings* exist where authors and reviewers collaborate to accept each other's papers [32]. Automated assignment techniques can raise the bar for dishonest collaborations considerably [28], yet our work shows that these techniques need to be implemented with care. Apart from collusion rings, dishonest authors can also work alone: They can try to promote an unfamiliar reviewer who might overlook paper issues and thus more likely submit a positive review.

We believe that dishonest authors more likely risk *deniable* manipulations such as a few spelling mistakes or additional references. Our evaluation shows this is sometimes already enough, for example, to promote an unfamiliar reviewer. As the line between adversarial and benign issues in a paper is often not clear, such an attack can be hard to discover. All in all, the automatic assignment of papers enables not only manipulations that undermine the entire reviewing process, but also small-scale attacks in which assignments are tweaked by a few deniable changes.

Table 3: Overview of related attacks against text classifiers.

	Perturbation				Constr.		Attack		
Paper	Char	Word	Sentence	Format	Semantics	Plausibility	Untargeted	Targeted	Classifier
This work	●	●	●	●	✓	✓	✓	✓	Assign.
Alzantot et al. [3]	●	●			✓		✓		NN
Ebrahimi et al. [18]	●	●			✓		✓		NN
Eger et al. [19]	●				✓		✓		NN
Gao et al. [21]	●				✓		✓		NN
Iyyer et al. [23]			●		✓		✓		NN
Jin et al. [25]		●			✓		✓		NN
Li et al. [29]	●	●			✓		✓		NN,LR
Liu et al. [33]	●	●			✓		✓		NN
Papernot et al. [43]		●					✓		NN
Ren et al. [49]		●			✓		✓		NN

6 Related Work

Our attack touches different areas of security research. In the following, we examine related concepts and methods.

Adversarial learning. A large body of work has focused on methods for creating adversarial examples that mislead learning-based systems [7]. However, most of this work considers attacks in the image domain and assumes a one-to-one mapping between pixels and features. This assumption does not hold in discrete domains, leading to the notion of *problem-space attacks* [45, 46]. Our work follows this research strand and introduces a new hybrid attack strategy for operating in both the feature space and problem space. Furthermore, we examine weak spots in *text preprocessing*, which extend the attack surface for adversarial papers. These findings complement prior work advocating that the security of preprocessing in machine learning needs be considered in general [47].

Table 3 summarizes prior work on misleading text classifiers. While we build on some insights developed in these works, text classification and paper assignment differ in substantial aspects: First, the majority of prior work focuses on untargeted attacks that aim at removing individual features. In our case, however, we have to consider a targeted attack where an adversary needs to specifically change the assignment of reviewers. Second, prior attacks often directly exploit the gradient of neural networks or compute a gradient by using word importance scores. Such gradient-style attacks are not applicable to probabilistic topic models.

In view of these differences, our work is more related to the attack from Zhou et al. [59] which studies the manipulation of LDA. The authors show that an evasion is *NP-hard* and present an attack to promote and demote individual LDA topics. For our manipulation, however, we need to adjust not only individual topics but the complete topic distribution as well as consider side effects with concurring reviewers.

Attacks on assignment systems. Finally, another strain of research has explored the robustness of paper-reviewer assignment systems. Most of these works are based on *content-masking attacks* [38, 56], which use format-level transformation to exploit the discrepancy between displayed and extracted text. More specifically, Markwood et al. [38] and Tran and Jaiswal [56], similar to our work, target the paper-reviewer assignment task. Their attack is evaluated against Latent Semantic Indexing [17]—that is not used in real-world systems like TPMS. Although Tran and Jaiswal [56] recognize the shortcomings of format-level transformations, they do not explore text-level transformations or the interplay between the problem space and feature space of topic models.

Complementary to our work, a further line of research focuses on the collusion of reviewers. These works have analyzed semi-automatic paper matching systems under the assumption that malicious researchers can manipulate the paper assignment by carefully adjusting their paper biddings. Jecmen et al. [24] propose a probabilistic matching to decrease the probability of a malicious reviewer to be assigned to a target submission, while Wu et al. [57] tries to limit the disproportional influence of malicious biddings.

7 Conclusion

In this paper, we demonstrate that current systems for automatic paper-reviewer assignments are vulnerable and can be misled by *adversarial papers*. On a broader level, we develop a novel framework for constructing adversarial examples in discrete domains through joint optimization in the problem space and feature space. Based on this framework, we can craft objects that satisfy real-world constraints and evade machine-learning models at the same time.

In summary, our work demonstrates a significant attack surface of current matching systems and motivates further security analysis prior to their deployment. As a result, we have informed the developers of TPMS and Autobid about our findings, as part of a responsible disclosure process.

Acknowledgments

We thank our shepherd and reviewers for their valuable comments and suggestions. We also thank Ajeeth Kularajan, Andreas Müller, Jonathan Evertz, and Sina Wette for their assistance as well as Charlotte Schwedes and Annabelle Walle for their support with the user study. This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2092 CASA – 390781972), the German Federal Ministry of Education and Research under the grant BIFOLD23B, and the European Research Council (ERC) under the consolidator grants MALFOY (101043410) and RS^3 (101045669). Moreover, this work was supported by a fellowship within the IFI program of the German Academic Exchange Service (DAAD) funded by the Federal Ministry of Education and Research (BMBF).

References

[1] The Most Common English Misspellings. Blogpost on Lexico, 2021.

[2] A. Agrawal, W. Fu, and T. Menzies. What is Wrong with Topic Modeling? And how to Fix it Using Search-Based Software Engineering. *Information and Software Technology*, 2018.

[3] M. Alzantot, Y. Sharma, A. Elgohary, B. Ho, M. B. Srivastava, and K. Chang. Generating Natural Language Adversarial Examples. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

[4] N. u. H. Bajwa, M. Langer, C. J. König, and H. Honecker. What Might get Published in Management and Applied Psychology? Experimentally Manipulating Implicit Expectations of Reviewers Regarding Hedges. *Scientometrics*, 2019.

[5] D. Balzarotti. System Security Circus. Post on personal blog, 2020.

[6] H. Bast. How Objective is Peer Review: The ESA Experiment. Blog post on CACM, 2018.

[7] B. Biggio and F. Roli. Wild patterns: Ten Years After the Rise of Adversarial Machine Learning. *Pattern Recognition*, 2018.

[8] S. Bird, E. Klein, and E. Loper. *Natural Language Processing with Python*. O'Reilly, 2009.

[9] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2002.

[10] N. Boucher, I. Shumailov, R. Anderson, and N. Papernot. Bad Characters: Imperceptible NLP Attacks. In *IEEE Symposium on Security and Privacy (S&P)*, 2022.

[11] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, and other. Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[12] G. Cabanac, C. Labbé, and A. Magazinov. Tortured Phrases: A Dubious Writing Style Emerging in Science. Evidence of Critical Issues Affecting Established Journals. *Computing Research Repository (CoRR)*, 2021.

[13] C. Carmony, X. Hu, H. Yin, A. V. Bhaskar, and M. Zhang. Extract Me If You Can: Abusing PDF Parsers in Malware Detectors. In *Symposium on Network and Distributed System Security (NDSS)*, 2016.

[14] L. Charlin and R. Zemel. The Toronto Paper Matching System: An Automated Paper-Reviewer Assignment System. In *International Conference on Machine Learning (ICML)*, 2013.

[15] S. Chaudhuri et al. Conference Management Toolkit (CMT).

[16] W. M. Darling. A Theoretical and Practical Implementation Tutorial on Topic Modeling and Gibbs Sampling. In *Annual Meeting of the Assoc. for Computational Linguistics: Human Language Technologies (HLT)*, 2011.

[17] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 1990.

[18] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou. HotFlip: White-Box Adversarial Examples for Text Classification. In *Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, 2018.

[19] S. Eger, G. G. Sahin, A. Rücklé, J. Lee, C. Schulz, M. Mesgar, K. Swarnkar, E. Simpson, and I. Gurevych. Text Processing Like Humans Do: Visually Attacking and Shielding NLP Systems. In *Conference of the North American Chapter of the Assoc. for Computational Linguistics: Human Language Technologies, (NAACL-HLT)*, 2019.

[20] T. Eisenhofer, L. Schönherr, J. Frank, L. Speckemeier, D. Kolossa, and T. Holz. Dompteur: Taming Audio Adversarial Examples. In *USENIX Security Symposium*, 2021.

[21] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi. Black-Box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers. In *IEEE Security and Privacy Workshops (SPW)*, 2018.

[22] M. D. Hoffman, D. M. Blei, and F. R. Bach. Online Learning for Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2010.

[23] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. In *Conference of the North American Chapter of the Assoc. for Computational Linguistics: Human Language Technologies, (NAACL-HLT)*, 2018.

[24] S. Jecmen, H. Zhang, R. Liu, N. B. Shah, V. Conitzer, and F. Fang. Mitigating Manipulation in Peer Review via Randomized Reviewer Assignments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[25] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[26] E. Kohler et al. HotCRP Conference Review Software.

[27] N. Lawrence and C. Cortes. The NIPS Experiment. Post on personal blog, 2014.

[28] K. Leyton-Brown, Mausam, Y. Nandwani, H. Zarkoob, C. Cameron, N. Newman, and D. Raghu. Matching Papers and Reviewers at Large Conferences. *Computing Research Repository (CoRR)*, 2022.

[29] J. Li, S. Ji, T. Du, B. Li, and T. Wang. TextBugger: Generating Adversarial Text Against Real-world Applications. In *Symposium on Network and Distributed System Security (NDSS)*, 2019.

[30] X. Li and T. Watanabe. Automatic Paper-to-reviewer Assignment, based on the Matching Degree of the Reviewers. In *International Conference in Knowledge Based and Intelligent Information and Engineering Systems (KES)*, 2013.

[31] H.-T. Lin, M. F. Balcan, R. Hadsell, and M. Ranzato. What we learned from NeurIPS 2020 reviewing process. Blog post on Medium, 2020.

[32] M. L. Littman. Collusion Rings Threaten the Integrity of Computer Science Research. *Communications of the ACM*, 2021.

[33] H. Liu, Y. Zhang, Y. Wang, Z. Lin, and Y. Chen. Joint Character-Level Word Embedding and Adversarial Stability Training to Defend Adversarial Text. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[34] X. Liu, T. Suel, and N. Memon. A Robust Model for Paper Reviewer Assignment. In *ACM Conference on Recommender Systems (RecSys)*, 2014.

[35] C. Long, R. C.-W. Wong, Y. Peng, and L. Ye. On Good and Fair Paper-Reviewer Assignment. In *IEEE International Conference on Data Mining (ICDM)*, 2013.

[36] J. B. Lovins. Development of a Stemming Algorithm. *Mechanical Translation and Computational Linguistics*, 1968.

[37] M. V. Mäntylä, M. Claes, and U. Farooq. Measuring LDA Topic Stability from Clusters of Replicated Runs. In *International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2018.

[38] I. Markwood, D. Shen, Y. Liu, and Z. Lu. PDF Mirage: Content Masking Attack Against Information-Based Online Services. In *USENIX Security Symposium*, 2017.

[39] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

[40] J. Mink, L. Luo, N. M. Barbosa, O. Figueira, Y. Wang, and G. Wang. DeepPhish: Understanding User Trust Towards Artificially Generated Profiles in Online Social Networks. In *USENIX Security Symposium*, 2022.

[41] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, and other. Training Language Models to Follow Instructions with Human Feedback. *Computing Research Repository (CoRR)*, 2022.

[42] N. Papernot, P. D. McDaniel, and I. J. Goodfellow. Transferability in Machine Learning: From Phenomena to Black-Box Attacks using Adversarial Samples. *Computing Research Repository (CoRR)*, 2016.

[43] N. Papernot, P. D. McDaniel, A. Swami, and R. E. Harang. Crafting Adversarial Input Sequences for Recurrent Neural Networks. In *IEEE Military Communications Conference (MILCOM)*, 2016.

[44] B. Parno. Autobid. Public GitHub Repository.

[45] F. Pierazzi, F. Pendlebury, J. Cortellazzi, and L. Cavallaro. Intriguing Properties of Adversarial ML Attacks in the Problem Space. In *IEEE Symposium on Security and Privacy (S&P)*, 2020.

[46] E. Quiring, A. Maier, and K. Rieck. Misleading Authorship Attribution of Source Code using Adversarial Learning. In *USENIX Security Symposium*, 2019.

[47] E. Quiring, D. Klein, D. Arp, M. Johns, and K. Rieck. Adversarial Preprocessing: Understanding and Preventing Image-Scaling Attacks in Machine Learning. In *USENIX Security Symposium*, 2020.

[48] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *LREC Workshop on New Challenges for NLP Frameworks*, 2010.

[49] S. Ren, Y. Deng, K. He, and W. Che. Generating Natural Language Adversarial Examples through Probability Weighted Word Saliency. In *Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, 2019.

[50] N. B. Shah. Challenges, Experiments, and Computational Solutions in Peer Review. *Communications of the ACM*, 2022.

[51] A. Soneji, F. B. Kokulu, C. Rubio-Medrano, T. Bao, R. Wang, Y. Shoshitaishvili, and A. Doupé. "Flawed, but like democracy we don't have a better system": The Experts' Insights on the Peer Review Process of Evaluating Security Papers. In *IEEE Symposium on Security and Privacy (S&P)*, 2022.

[52] I. Stelmakh, N. B. Shah, and A. Singh. PeerReview4All: Fair and Accurate Reviewer Assignment in Peer Review. In *Conference on Algorithmic Learning Theory (ALT)*, 2019.

[53] S. E. Sullivan, Y. Baruch, and H. Schepmyer. The Why, What, and How of Reviewer Education: A Human Capital Approach. *Journal of Management Education*, 2010.

[54] C. J. Taylor. On the Optimal Assignment of Conference Papers to Reviewers. Technical report, 2008.

[55] R. Taylor, M. Kardas, G. Cucurull, T. Scialom, A. Hartshorn, E. Saravia, A. Poulton, V. Kerkez, and R. Stojnic. Galactica: A Large Language Model for Science. *Computing Research Repository (CoRR)*, 2022.

[56] D. Tran and C. Jaiswal. PDFPhantom: Exploiting PDF Attacks Against Academic Conferences' Paper Submission Process with Counterattack. In *IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2019.

[57] R. Wu, C. Guo, F. Wu, R. Kidambi, L. van der Maaten, and K. Q. Weinberger. Making Paper Reviewing Robust to Bid Manipulation Attacks. In *International Conference on Machine Learning (ICML)*, 2021.

[58] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. T. Diab, X. Li, X. V. Lin, and other. OPT: Open Pre-trained Transformer Language Models. *Computing Research Repository (CoRR)*, 2022.

[59] Q. Zhou, H. Chen, Y. Zheng, and Z. Wang. EvaLDA: Efficient Evasion Attacks Towards Latent Dirichlet Allocation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

A Training Corpus

In the following, we describe how the corpus for the simulated paper-reviewer assignment process is generated. The PC of the *43rd IEEE Symposium on Security and Privacy* conference consists of 165 persons. For each PC member, we construct an archive of papers representative for the person's expertise and interests by crawling their *Google Scholar* profile. In rare cases, this profile is not available and we use the profile from *DBLP computer science bibliography* instead. We sort all papers first by year and then by number of citations to obtain an approximation of the recent research interests. From this list, we remove all papers with no citation and for which we cannot obtain a PDF file (e.g., paywalled files we cannot access). Furthermore, we remove papers that are already used as a target submission. From the remaining list, we select the first 40 papers (if available).

To construct reviewer archives A_r, we randomly sample 20 paper for each reviewer and compile the corpus as the union of these archives. The remaining 20 papers are used to simulate the black-box scenario. Here, we consider different levels of overlaps between 0% (i.e., no overlap between the training data of the surrogates and target system) and 100% (i.e., complete overlap).

B Hyperparameter Selection

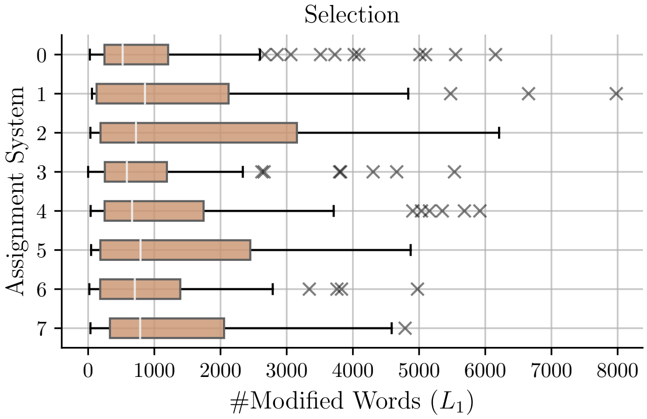
In the following, we describe how we determine the hyperparameters of our attack in the two scenarios.

White-box scenario. We perform a grid search over 100 randomly sampled targets from all three objectives and optimize parameters as a trade-off between attack efficacy and efficiency. To not overfit to a specific model, we train 8 AutoBid systems on different random seeds and randomly select one system per target. Note that an attacker with full-knowledge could also choose parameters that perform best for a specific target. We set the beam width $B = 8 \in \{2^0, \dots, 2^3\}$, step size $k = 128 \in \{2^5, \dots, 2^8\}$, number of successors $M = 512 \in \{2^7, \dots, 2^9\}$, and reviewer window to $\omega = 6 \in \{2^1, \dots, 2^3\}$ with offset $\upsilon = 3 \in \{0, \dots, 3\}$. The target rank for rejected reviewer is set to $rank_{\mathbf{x}}^{rej} = 10$ and we consider $\nu = 5000$ words per reviewer. We run the attack for at most $I = 1000$ iterations with at most $S = 8$ transitions between spaces and a target margin of $\gamma = -0.02$.

Black-box scenario. We repeat the grid search and train 8 systems on a surrogate corpus at 70% overlap. We randomly sample 100 targets from all three objectives and assign each a random surrogate system. We set the beam width $B = 4 \in \{2^0, \dots, 2^3\}$, step size $k = 256 \in \{2^5, \dots, 2^8\}$, number of successors $M = 128 \in \{2^7, \dots, 2^9\}$, and reviewer window to $\omega = 2 \in \{2^1, \dots, 2^3\}$ with offset $\upsilon = 1 \in \{0, \dots, 3\}$. Finally, to increase the robustness of our attack, we set the margin as $\gamma = -0.16$. All other parameters are the same as before.

C Feature-Space Search

We report the L_1 norms of individual attacks exemplary for the selection objective. We consider 8 different assignment system and sample 100 random targets per system (i. e., 800 attacks in total).



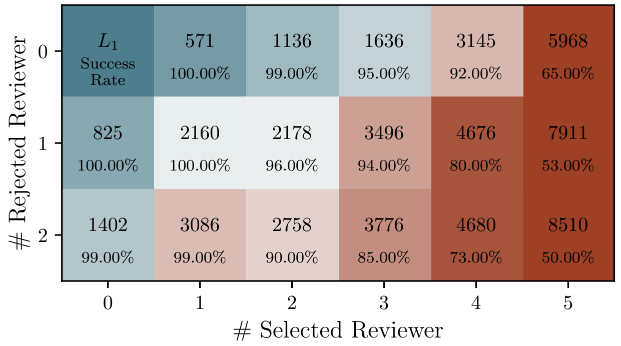
D Generalization of Attack

We empirically evaluate our attack on two conferences with differently sized committees: (a) the *29th USENIX Security Symposium* with 120 reviewers and (b) the *43rd IEEE Symposium on Security and Privacy* conference with a larger committee consisting of 165 reviewers. We simulate the attack for all three objectives and report the aggregated success rate, the median running time, and the median L_1 and L_∞ norm.

	Success Rate	Running Time	L_1	L_∞
USENIX ’20	99.62 %	7m 38s	1033	30
IEEE S&P ’22	99.67 %	7m 12s	1115	35

E Scaling of Target Reviewer

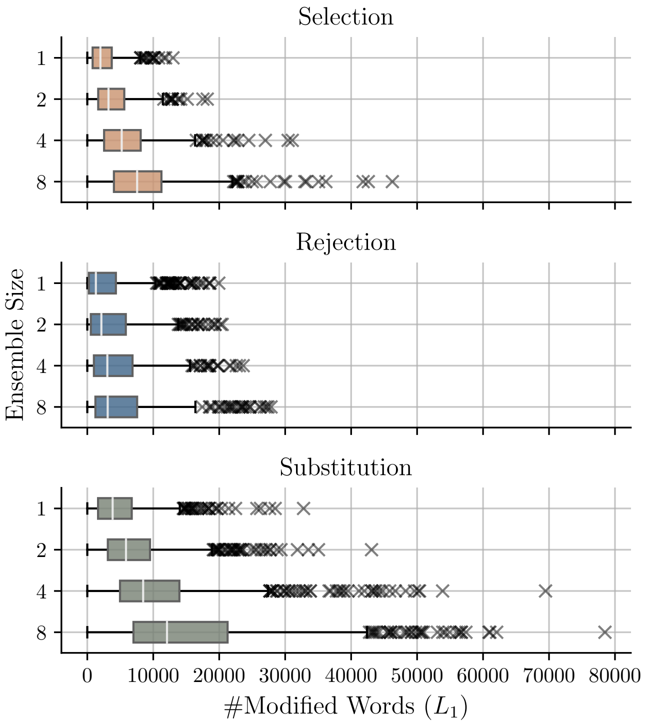
We run the attack for different combinations of the number of selected and rejected target reviewers. For each combination, we report the median L_1 norm as well as the success rate over 100 targets.



# Rejected Reviewer \ # Selected Reviewer	0	1	2	3	4	5
0	L_1 Success Rate	571 100.00%	1136 99.00%	1636 95.00%	3145 92.00%	5968 65.00%
1	825 100.00%	2160 100.00%	2178 96.00%	3496 94.00%	4676 80.00%	7911 53.00%
2	1402 99.00%	3086 99.00%	2758 90.00%	3776 85.00%	4680 73.00%	8510 50.00%

F Surrogate Ensembles

We report the L_1 norms for the black-box scenario with varying sizes of surrogate ensembles. We report the L_1 over 100 targets for all three objectives.



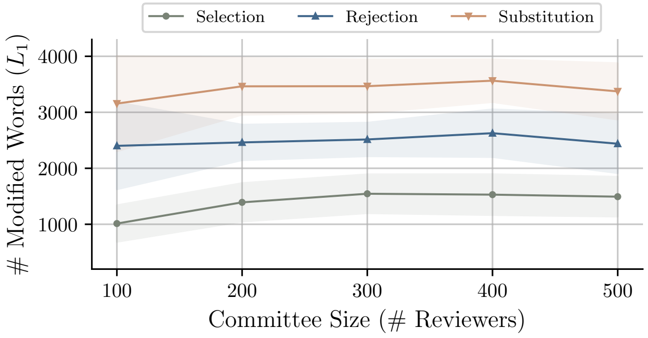
G Overlap

We compare the cross-entropy of reviewer-to-words distributions across models trained on a corpus with different overlaps. We randomly select 10 reviewers and report the mean cross-entropy and standard deviation between 8 models each (i.e., 64 pairs per overlap and reviewer).

#	Overlap			
	0%	30%	70%	100%
1	13.19 ± 0.46	13.13 ± 0.47	13.12 ± 0.37	13.20 ± 0.44
2	12.56 ± 0.29	12.55 ± 0.37	12.64 ± 0.34	12.50 ± 0.29
3	13.58 ± 0.63	13.56 ± 0.56	13.47 ± 0.62	13.52 ± 0.63
4	12.43 ± 0.50	12.29 ± 0.48	12.35 ± 0.54	12.32 ± 0.50
5	13.41 ± 0.51	13.41 ± 0.61	13.50 ± 0.56	13.31 ± 0.66
6	12.84 ± 0.23	12.81 ± 0.21	12.93 ± 0.25	12.90 ± 0.23
7	14.20 ± 0.42	14.28 ± 0.44	14.39 ± 0.48	14.08 ± 0.41
8	13.57 ± 0.46	13.59 ± 0.46	13.55 ± 0.40	13.66 ± 0.42
9	13.44 ± 0.72	13.33 ± 0.68	13.54 ± 0.67	13.44 ± 0.76
10	15.24 ± 0.59	15.08 ± 0.59	15.31 ± 0.66	14.88 ± 0.61

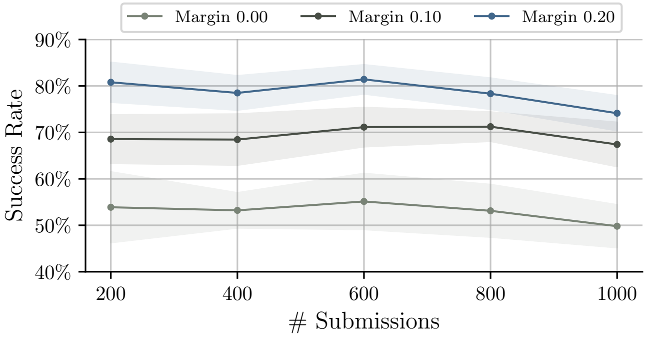
H Committee Size

We simulate the attack with varying sizes of the program committee. For each size, we report the mean number of required modifications over 8 target systems each sampled with a random committee. For each objective, we compute 280 adversarial papers per target system.



I Load balancing

We simulate the attack with varying numbers of concurring submissions between 200 and 1,000. We report the mean success rate over 8 target systems each sampled with a random committee. For each objective, we compute 280 adversarial papers per target system.



J Problem-Space Transformations

Detailed description of problem-space transformations. For the transformations' categorization, see Table 1.

Transformation	Description
Reference addition	Given a database of bibtex records of real papers, this transformation adds papers to the bibliography. It has two options: • *Add unmodified papers.* This option treats the insertion as optimization problem. It tries to find *k* bibtex records such that the number of added words is maximized. This allows us to maximize the impact of the added papers in the bibliography. • *Add adapted papers.* This option adds *r* words into a randomly selected bibtex record, which is then added to the bibliography. This transformation allows us to add very specific words which are difficult to add in the normal text in a meaningful way. In the experiments, *r* is set to 3, i. e., each added bibliography entry has only 3 additional words to avoid suspicion.
Synonym	This transformation replaces words by synonyms using a security domain-specific word embedding [39]. To this end, the word embeddings are computed on a collection of 11,770 security papers (Section 5 presents the dataset). Two options are implemented: • *Add.* Allows adding a word. For each word in the text, it obtains its synonyms. If one of the synonyms is in the list of words that should be added, the synonym is used as replacement for the text word. • *Delete.* Allows removing a word by replacing it with one of its synonyms. The transformation iterates over possible synonyms and only uses a synonym if it has the same part-of-speech (POS) tag as the original word. From the remaining set of synonyms, the transformation randomly chooses a candidate.
Spelling mistake	Inserts a spelling mistake into a word that should be deleted. • *Most common misspelling.* This option tries to find a misspelling from a list of 78 rules for most common misspellings, such as *appearence* instead of *appearance* (rule: ends with -ance), or *basicly* instead of *basically* (rule: ends with -ally). • *Swap or delete.* Swap two adjacent letters or delete a letter in the word. Chooses between both ways randomly. The transformation first tries to find a common misspelling, and if not possible, it applies the swap-or-delete strategy.
Language model	Uses a language model, here OPT [58], to create sentences with the requested words. To create more security-related sentences, we use the corpus from Section 5 consisting of 11,770 security papers to finetune the OPT-350m model. Equipped with this model, the transformer appends new text at the end of the related work or discussion section. To this end, we extract some text before the insertion position and ask the model to complete the text while choosing suitable words from the set of requested words.
Homoglyph	Replaces a single character in a word by a visually identical or similar homoglyph. For instance, we can replace the Latin letter *A* by its Cyrillic equivalent.
Hidden box	Uses the accessibility support with the latex package *accsupp* that allows defining an alternative text over an input text. Only the input text is visible, while the feature extractor processes the alternative text. This allows adding an arbitrary number of words as alternative text. As the input text is not processed, we can also delete words or text in this way. Two options are implemented: • *Add.* Allows adding an arbitrary number of words in the alternative text. This step requires defining the alternative text at least over a visible word that is, however, not extracted as feature afterwards anymore. To reduce side effects, the transformation first checks if the attack requests a word to be reduced. If so, it lays the alternative text over this word. Otherwise, a stop word is chosen that would be ignored in the preprocessing stage anyway. The step thus reduces possible side effects. • *Delete.* Adds an empty alternative text over the input word that needs to be removed, so that the word is not extracted anymore.