REED: Post-Training Representation Editing for Cross-Domain Linguistic Steganalysis

Anonymous ACL submission

Abstract

In real-world scenarios of linguistic steganalysis, tested texts usually come from unseen domains with different vocabularies, topics, writing styles, and steganographic generation patterns, which can significantly degrade the detection performance. Although existing cross-domain steganalysis methods can effectively alleviate this problem through distribution alignment, domain-invariant feature learning, etc., the detection performance is not satisfactory. In this paper, we propose a post-training representation editing method for cross-domain linguistic steganalysis. Specifically, the detector is first trained on source-domain data, and then the feature extractor and classifier are kept frozen, and the intermediate representations are deterministically edited before classification. For domain adaptation, we construct a domain-offset vector from marginal source and target representations. For domain generalization, we derive a source-domain cover-to-stego direction to guide sample-specific editing. Experimental results show that compared with the advanced methods, the proposed method can achieve high cross-domain detection performance, especially in terms of F1-score, while requiring no architecture modification or parameter updates after source-domain training.

1 Introduction

Linguistic steganography embeds secret messages into natural text, enabling covert communication over public channels (Anderson and Petitcolas, 1998). Recent generation-based methods, driven by neural language models, can produce fluent stego texts with high embedding capacity and strong imperceptibility (Ziegler et al., 2019; Zhang et al., 2021; Zhou et al., 2022). While this progress improves the practicality of steganography, it also increases the risk of abusing public text channels to transmit hidden information (Mazurczyk and Wendzel, 2018). Linguistic steganalysis therefore aims to distinguish stego texts from normal cover texts and has become an important countermeasure.

Existing steganalysis methods mainly rely on deep neural detectors. including convolutional neural networks (CNNs), recurrent neural networks (RNNs), graph-based modeling, hierarchical learning, social-network-oriented detection, and attention-based feature fusion(Wen et al., 2019; Yang et al., 2019b; Niu et al., 2019; Peng et al., 2021; Wu et al., 2021; Xue et al., 2022a; Yang et al., 2023; Peng et al., 2023). These methods achieve promising results under the independent identically distributed (i.i.d.) assumption. In practice, however, this assumption is often violated. Suspicious texts may come from domains different from the training data and may differ in vocabulary, topic, length, and writing style, which can severely degrade the performance of a source-trained detector (Xue et al., 2024; Wu et al., 2024).

Similar to other classification tasks, cross-domain linguistic steganalysis focuses on two mismatch problems, containing the domain adaptation and the domain generalization (Xue et al., 2022b, 2024; Luo et al., 2025; Yang et al., 2025; Wu et al., 2024; Wang et al., 2021). For domain adaptation, learning domain-invariant features and adversarial learning are the commonly used paradigms. For domain generation, the effectiveness of the test-time generalization has been explored. Although the above methods improve the detection performance, these methods cannot achieve satisfactory performance, especially for domain generalization. The main reason is that once a detector is trained on the source domain, its classifier fixes the decision boundary in the learned representation space, and the target-domain representations are shifted away from the source distribution and become mismatched with this boundary (Xue et al., 2024; Yang et al., 2025). Moreover, they usually require additional optimization objectives, target-domain fine-tuning, or test-time parameter updates, which re-

quire high computational cost.

This above analysis leads to an interesting question: ***instead of parameter modification on target domain data, can we directly edit the representations before classification?***

In this paper, we propose a lightweight representation editing framework, called ***REED***, for cross-domain linguistic steganalysis. The steganographic detector is first trained on source domain data, and then each test representation is edited by a task-dependent vector before being passed to the classifier. For domain adaptation, the vector is estimated from the marginal representation offset between the source and target domains. For domain generalization, where target samples are unavailable before testing, the editing direction is derived from the source-domain cover-stego representation difference. The proposed REED operates directly in the learned feature space, without modifying the architecture, introducing extra training losses, or updating model parameters after source-domain training. Experimental results show that the REED improves cross-domain detection performance for both domain adaptation and generalization tasks.

2 Related Work

2.1 Linguistic Steganography

Linguistic steganography aims to embed secret information into natural text while hiding the existence of communication (Anderson and Petitcolas, 1998; Ziegler et al., 2019). Early studies mainly follow the modification-based paradigm, where secret bits are embedded by modifying an existing cover text, such as through synonym substitution or format changes (Cao et al., 2013; Xiang et al., 2014; Shirali-Shahreza, 2008). Since natural text has limited redundancy, these methods often suffer from restricted embedding capacity and may affect text naturalness.

With the development of neural language models, generation-based linguistic steganography has become a major direction (Fang et al., 2017; Yang et al., 2019a; Ziegler et al., 2019; Zhang et al., 2021). Instead of modifying a given text, these methods generate stego text directly by controlling word or token selection during decoding. Early neural methods use language models to map secret bits to candidate words, while later coding-based methods further exploit the probability distribution of the language model. For example, Huffman Coding assigns variable-length codes according to token probabilities, and Arithmetic Coding maps secret bits to probability intervals to improve embedding efficiency (Yang et al., 2018; Ziegler et al., 2019). Adaptive Dynamic Grouping further partitions candidate tokens according to their probabilities, reducing the distortion introduced during generation (Zhang et al., 2021). More recent methods such as Meteor and iMEC study stronger security guarantees for generative steganography (Kaptchuk et al., 2021; Schroeder de Witt et al., 2023). These advances make stego text increasingly fluent and difficult to detect, which in turn raises the need for reliable linguistic steganalysis.

2.2 Linguistic Steganalysis

To counter the development of linguistic steganography, existing linguistic steganalysis methods mainly employ deep learning models to learn discriminative features between cover and stego texts. Early neural approaches use CNNs, RNNs, or hybrid architectures to capture textual patterns for cover-stego classification (Wen et al., 2019; Yang et al., 2019b; Niu et al., 2019). Recent work has further improved feature learning for linguistic steganalysis. Fine-tuned pre-trained language models are used to capture more subtle inconsistencies in generated text, while graph-based modeling, hierarchical learning, and attention mechanisms have also been explored to enhance steganalysis representations (Devlin et al., 2019; Peng et al., 2021; Wu et al., 2021; Xue et al., 2022a; Peng et al., 2023). In addition, some studies consider more realistic detection scenarios, such as linguistic steganalysis on social networks (Yang et al., 2023).

2.3 Cross-Domain Linguistic Steganalysis

To alleviate the performance degradation caused by domain mismatch, existing studies mainly resort to domain adaptation to improve the model's transferability to the target domain (Xue et al., 2022b, 2024; Luo et al., 2025; Yang et al., 2025). MDA (Xue et al., 2022b) first introduces domain adaptation and transductive learning into text steganalysis, where the distributions of source and target features are aligned with Maximum Mean Discrepancy (MMD) (Gretton et al., 2012). SANet (Xue et al., 2024) further designs a Steganographic Domain Distance Metric (SDDM) and an adaptive weight selection network to measure source–target discrepancy and enhance domain-invariant feature extraction. PDTS (Luo et al., 2025) adopts a two-stage framework, which first pre-trains the model

on labeled source-domain data and then fine-tunes it with pseudo-labels from the target domain to capture target-domain-specific features. More recently, CADA (Yang et al., 2025) introduces class-aware adversarial pre-training and class-aware fine-tuning, where class-aware alignment and class-balanced pseudo-labels are used to improve both transferability and discrimination.

Beyond domain adaptation, domain generalization considers a more restrictive setting, where the model is expected to generalize to unseen target domains without accessing target-domain data during training (Zhou et al., 2023). In this context, test-time adaptation has been explored as a practical way to improve out-of-distribution prediction by adapting a trained model using test data only (Sun et al., 2020; Wang et al., 2021; Niu et al., 2022). TTALS (Wu et al., 2024) brings this idea into cross-domain linguistic steganalysis and updates the model during testing by minimizing the prediction entropy (Wang et al., 2021).

Overall, existing cross-domain methods usually adapt the model through additional alignment losses, pseudo-label-based fine-tuning, or test-time parameter updates (Xue et al., 2024; Yang et al., 2025; Wu et al., 2024). In contrast, our method keeps the source-trained model fixed and directly edits intermediate representations after training. It therefore provides a lightweight post-training alternative that operates in feature space without changing the model architecture or updating model parameters.

3 Method

3.1 Problem Formulation

In domain adaptation, labeled source-domain texts and unlabeled target-domain texts are available (Xue et al., 2022b, 2024; Yang et al., 2025). We denote them as

$$\mathcal{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}, \quad \mathcal{D}_t = \{x_j^t\}_{j=1}^{N_t}, \tag{1}$$

where x_i^s and x_j^t are source- and target-domain texts, respectively, and $y_i^s \in \{0, 1\}$ denotes the source label, with 0 for cover and 1 for stego. Target-domain labels are unavailable during training.

In domain generalization, only the labeled source-domain dataset $\mathcal{D}_s$ is available before testing, while target-domain texts and labels are inaccessible (Wu et al., 2024). This setting is more restrictive, since the detector must generalize to unseen domains without observing target-domain samples in advance.

We denote the source-trained steganalysis model as $h(x) = g_\phi(f_\theta(x))$, where f_θ is the feature extractor and g_ϕ is the classifier. The intermediate representation produced by f_θ is denoted by z.

3.2 Vector-Guided Representation Editing

The overall framework of the proposed method is illustrated in Figure 1.

We propose a vector-guided representation editing framework for cross-domain linguistic steganalysis. Unlike methods that introduce auxiliary adaptation losses (Xue et al., 2022b, 2024; Yang et al., 2025) or update model parameters at test time (Wu et al., 2024; Wang et al., 2021), our method keeps the source-trained model frozen and edits intermediate representations before classification.

The framework contains three stages. First, a base steganalysis model is trained on labeled source-domain data following the adopted backbone, including its feature extractor, classifier, trainable-parameter setting, and training protocol. No target-domain samples or extra adaptation losses are used in this stage. Second, after source-domain training, all parameters are frozen and editing vectors are constructed from intermediate representations. In domain adaptation, the vector is estimated from source and unlabeled target representations; in domain generalization, it is derived from source-domain cover and stego representations. Third, during testing, each input representation is edited before being passed to the frozen classifier:

$$z = f_\theta(x), \quad z \in \mathbb{R}^d, \tag{2}$$

$$\tilde{z} = z + \Delta z, \tag{3}$$

$$\hat{y} = \arg\max_{c \in \{0,1\}} g_\phi(\tilde{z})_c. \tag{4}$$

Here, Δz is constructed differently under domain adaptation and domain generalization, as described below.

3.3 Domain-Offset Representation Editing for Domain Adaptation

In the domain adaptation setting, unlabeled target-domain texts allow us to estimate the source-to-target shift in the learned representation space. We approximate this shift with a global domain-offset vector. Given M_s source-domain texts $\{x_i^s\}_{i=1}^{M_s}$ and M_t target-domain texts $\{x_j^t\}_{j=1}^{M_t}$ for vector estimation, we compute their mean representations with

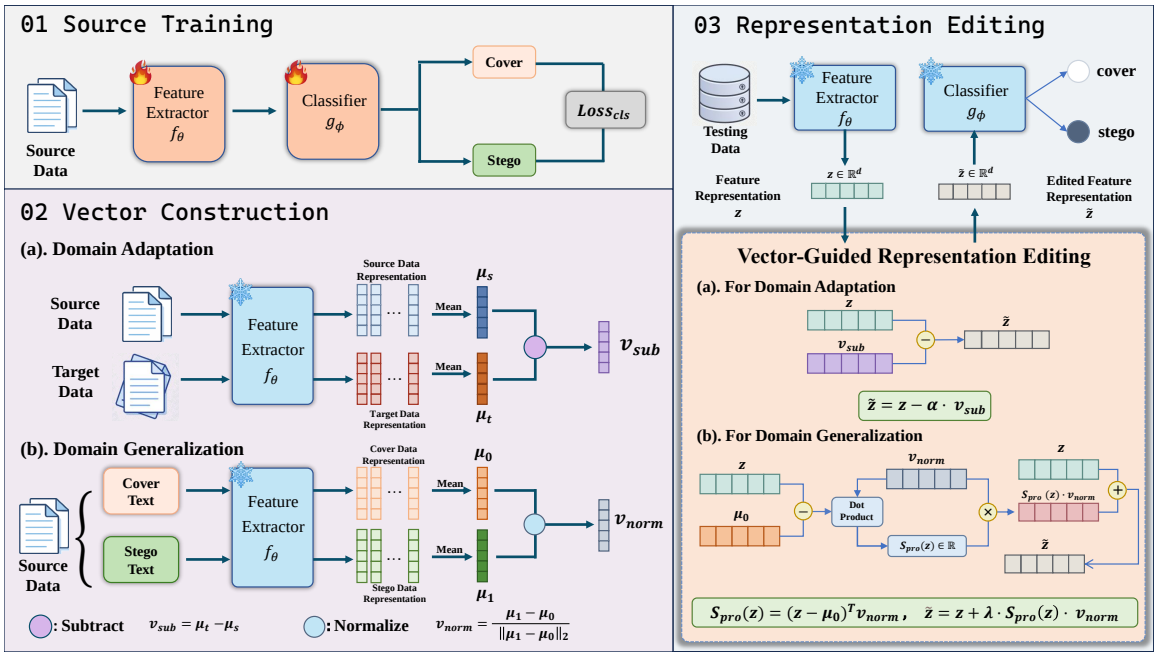

Figure 1: Overview of the proposed REED framework. After source-domain training, intermediate representations are edited by task-specific vectors before classification.

the frozen feature extractor:

$$\mu_s = \frac{1}{M_s}\sum_{i=1}^{M_s} f_\theta(x_i^s), \quad \mu_t = \frac{1}{M_t}\sum_{j=1}^{M_t} f_\theta(x_j^t). \tag{5}$$

We define the subtraction vector as

$$v_{\text{sub}} = \mu_t - \mu_s, \quad v_{\text{sub}} \in \mathbb{R}^d. \tag{6}$$

For a target-domain test sample x with representation $z = f_\theta(x)$, we edit its representation by

$$\tilde{z} = z - \alpha v_{\text{sub}}, \tag{7}$$

where α controls the editing strength. Unless otherwise specified, we set $\alpha = 1$ in our experiments. When target representations are globally shifted from the source distribution, subtracting v_{sub} provides a direct editing toward the representation space learned by the source-trained classifier.

3.4 Steganographic-Direction Representation Editing for Domain Generalization

In the domain generalization setting, target-domain samples are unavailable before testing, so the source-to-target offset cannot be estimated. Instead, we derive a cover-to-stego direction from source-domain representations to guide the editing of unseen-domain representations.

Let

$$\mathcal{D}_s^0 = \{x_i^0\}_{i=1}^{N_0}, \quad \mathcal{D}_s^1 = \{x_i^1\}_{i=1}^{N_1} \tag{8}$$

denote source-domain cover and stego texts, respectively. Using the frozen feature extractor, we compute their mean representations as

$$\mu_0 = \frac{1}{N_0}\sum_{i=1}^{N_0} f_\theta(x_i^0), \quad \mu_1 = \frac{1}{N_1}\sum_{i=1}^{N_1} f_\theta(x_i^1). \tag{9}$$

The normalized cover–stego direction is defined as

$$v_{\text{norm}} = \frac{\mu_1 - \mu_0}{\|\mu_1 - \mu_0\|_2}, \quad v_{\text{norm}} \in \mathbb{R}^d. \tag{10}$$

For a test sample x, its representation z is projected onto this source-derived direction:

$$S_{\text{pro}}(z) = (z - \mu_0)^\top v_{\text{norm}}, \quad S_{\text{pro}}(z) \in \mathbb{R}. \tag{11}$$

The projection score measures the signed displacement of the test representation z from the source cover mean along the cover–stego direction. We then obtain the edited representation $\tilde{z}$ by

$$\tilde{z} = z + \lambda S_{\text{pro}}(z) v_{\text{norm}}, \tag{12}$$

where λ controls the editing strength. The edited representation $\tilde{z}$ is classified by the frozen classifier as in Eq. (4).

This editing is sample-specific because $S_{\text{pro}}(z)$ is computed for each test representation. Unlike the DA vector, which applies a fixed translation to target samples, the DG editing is modulated by each

Table 1: Statistics of Cover and Stego Text Datasets for Different Domains and Steganography Algorithms

Method	News		Twitter		Movie	
	AL	BPW	AL	BPW	AL	BPW
Cover	19.13	–	8.02	–	18.94	–
AC	15.09	3.50	10.77	3.45	18.18	3.47
HC	22.29	3.71	11.74	3.69	19.41	3.64
ADG	22.75	5.41	10.90	5.13	20.66	5.24
Meteor	21.37	2.58	9.69	2.45	16.47	2.60
iMEC	23.18	3.77	11.09	3.70	19.56	3.89

sample's projection on the source-derived cover–stego direction. Since different steganographic algorithms may induce representation shifts with different magnitudes, we use an algorithm-level coefficient λ in the experiments. For each steganographic algorithm, the same coefficient is fixed across all source–target transfer directions.

4 Experiments

4.1 Basic Experimental Setup

Datasets and steganographic methods. We evaluate our method on three text corpora: Twitter (Go et al., 2009), Movie (Maas et al., 2011), and News (Thompson, 2017). Five linguistic steganographic methods are considered, including Arithmetic Coding (AC) (Ziegler et al., 2019), Huffman Coding (HC) (Yang et al., 2018), Adaptive Dynamic Grouping (ADG) (Zhang et al., 2021), Meteor (Kaptchuk et al., 2021), and iMEC (Schroeder de Witt et al., 2023). For each steganographic method, we construct six directed cross-corpus steganalysis tasks among the three corpora. Following the notation used in cross-domain steganalysis experiments, T, M, and N denote Twitter, Movie, and News, respectively. Thus, T-M, T-N, N-M, M-T, N-T, and M-N denote Twitter→Movie, Twitter→News, News→Movie, Movie→Twitter, News→Twitter, and Movie→News, respectively. For AC, HC, and ADG, stego texts are generated using LSTM language models trained on each corpus. For Meteor and iMEC, we use GPT-2 models (Radford et al., 2019) fine-tuned on the corresponding corpus for stego-text generation. The average length (AL) and bits per word (BPW) of the generated texts are reported in Table 1.

Data construction. The cover texts are randomly sampled from the original corpora. Following previous cross-domain linguistic steganalysis studies, for each corpus and each steganographic method, the training set contains 10,000 cover texts and 10,000 stego texts. The validation set and the test set each contain 1,000 cover texts and 1,000 stego texts. The cover and stego samples are kept balanced in all splits. In addition, the training, validation, and test sets are strictly separated, and no text sample is allowed to appear in more than one split.

Evaluation metrics. We use Accuracy (Acc) and F1-score (F1) as the evaluation metrics. The stego texts are treated as positive samples. The two metrics are defined as follows:

$$\mathrm{Acc} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{13}$$

$$\mathrm{F1} = \frac{2TP}{2TP + FP + FN}, \tag{14}$$

where TP, TN, FP, and FN denote true positives, true negatives, false positives, and false negatives, respectively.

Implementation details. For the Domain Adaptation experiments, we follow the training hyperparameters of the corresponding backbone methods, SANet (Xue et al., 2024) and CADA (Yang et al., 2025). Following the balanced benchmark protocol used in previous cross-domain linguistic steganalysis studies, the domain-offset vector is estimated from 1,000 source-domain training samples and 1,000 target-domain samples. The target-domain samples are used to estimate only the marginal target-domain representation mean, rather than class-conditional target statistics. No target-domain labels are used by the proposed editing algorithm for classification loss, pseudo-label generation, class-conditional estimation, class-conditional optimization, or parameter updates. The vector coefficient is fixed to 1.

For the Domain Generalization experiments, we fully fine-tune BERT (Devlin et al., 2019) and the classifier on the labeled source-domain training set with a learning rate of 2×10^{-5} for 3 epochs. The steganographic direction vector is estimated from 2,000 source-domain samples, including 1,000 cover texts and 1,000 stego texts. Since different steganographic algorithms may induce representation shifts with different magnitudes, we conduct an algorithm-level coefficient search for λ within a predefined range. For each steganographic algorithm, the selected coefficient is fixed across all six source–target transfer directions, rather than tuned separately for each transfer task. The detailed coefficients are reported in Appendix A.

Table 2: Detailed Results on the Domain Adaptation Task

Algorithm	Model	T-M	T-N	N-M	M-T	N-T	M-N	Avg.
AC	SANet	73.50/70.69	74.95/69.73	**73.85/69.58**	72.10/67.56	59.90/49.18	81.65/79.04	72.66/67.63
	SANet+REED	**76.20/76.87**	**77.70/77.72**	70.80/68.30	**73.70**/71.96	60.05/58.92	82.15/81.78	**73.43/72.59**
	CADA	65.65/67.86	58.95/61.14	68.85/65.97	65.50/68.31	55.78/**59.23**	**84.70/83.44**	66.57/67.66
	CADA+REED	73.35/73.80	76.15/75.77	63.60/56.24	71.70/**76.53**	**61.60**/54.72	79.20/80.97	70.93/69.67
HC	SANet	74.80/75.75	**66.10**/58.30	**78.75**/77.19	**74.20/74.78**	72.50/69.72	**72.00**/65.22	**73.06**/70.16
	SANet+REED	**76.70/78.33**	64.85/**64.90**	**78.75/78.97**	72.95/74.08	72.90/**73.06**	67.80/**68.31**	72.33/**72.94**
	CADA	72.90/72.43	64.20/55.25	76.25/73.03	69.45/64.37	69.15/61.37	66.20/53.69	69.69/63.36
	CADA+REED	74.00/74.93	64.10/63.85	78.50/78.63	71.70/72.44	**72.95**/72.80	68.00/67.81	71.54/71.74
ADG	SANet	57.25/57.23	56.55/56.36	65.25/65.24	61.37/60.65	61.13/59.62	64.70/64.62	61.04/60.62
	SANet+REED	62.85/62.33	**62.90/60.36**	**72.75/72.38**	**64.30/62.72**	**65.25**/58.31	**73.05/70.66**	**66.85/64.46**
	CADA	52.60/52.59	54.36/54.31	52.31/52.15	53.39/53.35	52.95/52.69	53.74/53.67	53.23/53.13
	CADA+REED	**66.40/63.48**	62.25/58.95	69.55/67.13	62.60/58.86	64.90/**63.40**	69.35/65.43	65.84/62.88
Meteor	SANet	74.55/77.43	66.45/67.47	77.20/78.02	**77.55**/77.49	**73.30**/71.20	**79.50**/78.67	74.76/75.05
	SANet+REED	**77.70/79.10**	**68.45/69.95**	**79.10/80.11**	75.20/77.56	72.50/72.69	77.50/**79.24**	**75.08/76.44**
	CADA	65.32/66.28	60.05/63.23	69.10/72.06	64.50/67.42	68.40/72.49	69.80/73.57	66.19/69.17
	CADA+REED	74.05/76.74	62.35/65.16	76.60/76.56	76.00/**78.59**	72.20/**73.14**	76.60/77.61	72.97/74.63
iMEC	SANet	59.00/**63.35**	57.75/53.59	67.75/67.74	**71.20/73.43**	66.70/65.89	67.60/72.11	65.00/66.02
	SANet+REED	**61.85**/58.10	**59.35**/52.92	**70.65/79.22**	69.80/67.56	66.25/65.61	**73.95**/72.73	**66.98**/66.02
	CADA	58.37/57.08	56.27/53.49	66.64/71.25	70.34/68.45	65.36/63.40	65.53/70.89	63.75/64.09
	CADA+REED	57.70/61.93	58.49/**53.72**	**70.65**/70.52	68.70/70.21	**68.40/69.20**	72.65/**73.64**	66.10/**66.54**

For comparison with a parameter-update-based test-time adaptation baseline under the same source-trained setting, we apply a TTALS-style entropy-minimization update (Wu et al., 2024) to the source-trained BERT-FT model, denoted as BERT-FT+TTem. This differs from the original TTALS setting, which performs fully test-time adaptation from a pre-trained language model without source-domain training. For BERT-FT+TTem, we use a learning rate of 1×10^{-5} and perform one entropy-minimization step (Wang et al., 2021) for each target-domain batch.

4.2 Results on Domain Adaptation

We first evaluate the proposed method under the Domain Adaptation task by applying it to two representative cross-domain steganalysis methods, SANet (Xue et al., 2024) and CADA (Yang et al., 2025). For a fair comparison, we keep the same feature extractor, classifier, and training strategy as each backbone method. After the backbone model is trained, we compute the domain-offset vector from the marginal source and target representation means and apply it to the learned representations during testing. Table 2 reports the Acc and F1 results on six directed cross-corpus tasks, with the last column showing the average performance over all tasks.

As shown in Table 2, adding the domain-offset vector improves the overall performance of both SANet and CADA on most cross-domain tasks. For SANet, the average Acc increases from 69.30% to 70.93%, and the average F1 increases from 67.90% to 70.49%. For CADA, the improvement is more pronounced, with the average Acc increasing from 63.89% to 69.48% and the average F1 increasing from 63.48% to 69.09%. These results indicate that our method is not tied to a specific backbone, but can be used as a post-training representation editing strategy for different cross-domain steganalysis models. The improvement on ADG is particularly notable. CADA+REED improves the average Acc/F1 from 53.23%/53.13% to 65.84%/62.88%, suggesting that when the original backbone still suffers from a clear source-target representation mismatch, the proposed vector editing can effectively compensate for such mismatch in the feature space.

It is worth noting that our experiments are mainly conducted under relatively high embedding-rate settings. In such settings, steganographic traces are generally more detectable, but they may also become more entangled with corpus-specific generation patterns and textual distributions, which makes cross-domain alignment more challenging. Although SANet and CADA reduce domain discrepancy through training-time alignment, they need to balance domain alignment with cover–stego dis-

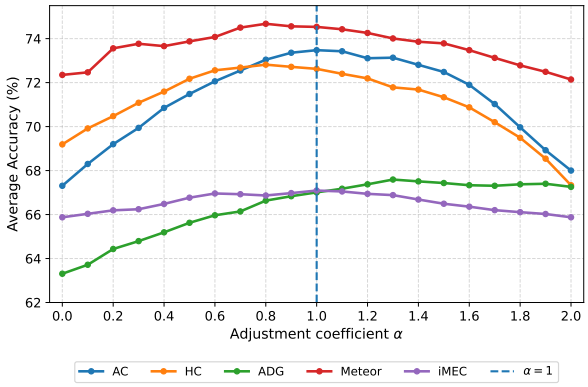

Figure 2: Sensitivity of the editing strength α in domain adaptation.

crimination. When steganographic variations are strong, this joint optimization may still leave residual source–target offsets or weaken class separation. By contrast, our method performs post-training representation editing. After the backbone has learned discriminative steganographic features, the estimated domain-offset vector directly compensates for the remaining source–target displacement before classification. In this way, vector editing can reduce residual domain mismatch while better preserving the learned cover–stego structure, which helps explain its effectiveness under high embedding-rate settings.

4.3 Ablation Study on the Editing Strength in Domain Adaptation

We further analyze the effect of the editing strength α in the Domain Adaptation setting. Here, α controls the magnitude of the domain-offset vector applied to target-domain representations. When $\alpha = 0$, no vector editing is performed, while larger values apply stronger representation editing along the estimated source–target offset direction.

As shown in Figure 2, the average accuracy generally increases as α grows from 0 to around 1, indicating that the estimated domain-offset vector effectively reduces the representation mismatch between the source and target domains. However, further increasing α does not consistently bring additional gains and may even lead to performance degradation. This suggests that overly strong editing may move target representations beyond the appropriate editing range.

Overall, setting $\alpha = 1$ provides a stable and effective choice across different steganographic algorithms. This observation suggests that the estimated domain-offset vector already captures not only the editing direction but also a reasonable scale of the source–target displacement in the representation space. Therefore, our method does not require heavy tuning of the editing strength, and we fix $\alpha = 1$ in all Domain Adaptation experiments.

4.4 Results on Domain Generalization

We further evaluate the proposed method under the Domain Generalization task. In this setting, we first fully fine-tune a BERT-based classifier (Devlin et al., 2019) on the labeled source domain. To ensure a fair comparison, all methods start from the same source-trained BERT-FT model. The source-trained model is directly used as the source-only BERT-FT baseline, while BERT-FT+TTem applies the entropy-minimization update of TTALS (Wu et al., 2024) to this model. This setting differs from the original fully test-time adaptation setting of TTALS, but allows us to compare parameter-update-based test-time adaptation and our post-training representation editing under the same source-trained initialization. Table 3 reports the Acc and F1 results on six directed cross-corpus tasks, with the last column showing the average performance over all tasks.

As shown in Table 3, our method improves the source-only BERT-FT baseline on most steganographic algorithms, especially in terms of F1-score. Averaged over all evaluated algorithms and transfer directions, our method improves Acc from 70.73% to 71.87%, and F1 from 66.06% to 73.36%. The improvement in F1 indicates that the source-derived steganographic direction helps reduce the imbalance between cover and stego detection on unseen target domains. The gains are especially notable on AC and HC. For AC, the average Acc/F1 increases from 69.94%/58.66% to 74.80%/72.47%. For HC, the average Acc/F1 increases from 70.68%/62.02% to 76.94%/76.82%. These results suggest that, once the editing coefficient is fixed, the cover–stego direction estimated from the source domain can provide useful guidance for target-domain representation editing.

Compared with TTALS, our method avoids test-time parameter updates and therefore shows more stable behavior in many tasks. TTALS degrades performance on several transfer directions, especially in terms of F1. This may be because entropy minimization updates model parameters according to the model's own predictions on target-domain samples. When the source-trained model produces biased or unreliable predictions, the entropy ob-

Table 3: Detailed Results on the Domain Generalization Task

Algorithm	Model	T-M	T-N	N-M	M-T	N-T	M-N	Avg.
AC	BERT-FT	79.45/78.98	79.50/76.60	50.10/20.95	72.50/67.34	55.20/26.46	82.90/81.61	69.94/58.66
	BERT-FT+TTem	**80.40**/80.06	61.60/39.14	50.95/3.91	53.10/13.79	50.95/4.26	82.50/79.82	63.25/36.83
	BERT-FT+REED	78.75/**81.64**	**81.95/82.31**	**62.95/47.93**	**77.10/79.80**	**63.70/58.94**	**84.35/84.22**	**74.80/72.47**
HC	BERT-FT	77.70/76.05	64.15/50.59	74.40/67.43	71.75/65.14	67.00/53.59	69.10/59.32	70.68/62.02
	BERT-FT+TTem	64.20/73.04	52.65/12.74	51.65/6.75	55.00/18.77	50.95/4.11	54.45/16.65	54.82/22.01
	BERT-FT+REED	**79.15/81.75**	**68.55/64.72**	**82.35/82.65**	**77.00/79.11**	**77.90/78.46**	**76.70/74.25**	**76.94/76.82**
ADG	BERT-FT	68.30/62.97	65.75/57.37	73.30/68.33	66.50/60.77	**67.45**/65.83	74.55/68.68	69.31/63.99
	BERT-FT+TTem	51.50/8.83	53.25/15.23	56.25/23.85	**66.85**/70.01	57.85/69.93	54.25/16.29	56.66/34.02
	BERT-FT+REED	**69.25/71.59**	**66.20/65.10**	**73.85/75.84**	66.35/**72.22**	63.45/**71.46**	**77.80/78.19**	**69.48/72.40**
Meteor	BERT-FT	**74.40/78.98**	75.05/77.99	**79.40/80.40**	**76.15**/75.01	**75.15**/74.18	**81.00**/79.30	**76.86**/77.64
	BERT-FT+TTem	68.40/75.91	66.60/74.35	62.10/41.15	73.75/77.95	62.50/47.41	59.15/33.63	65.42/58.40
	BERT-FT+REED	69.95/74.50	**77.65/80.52**	73.80/78.24	75.40/**79.67**	70.20/**76.06**	76.95/**80.54**	73.99/**78.26**
iMEC	BERT-FT	55.89/67.23	60.78/65.45	**69.89/65.41**	**70.02**/67.96	**71.28**/65.92	**73.19/75.98**	**66.84/67.99**
	BERT-FT+TTem	49.76/50.66	50.19/53.41	57.56/55.74	65.13/58.26	50.29/55.13	67.98/60.58	56.82/55.63
	BERT-FT+REED	**56.07/68.93**	**62.13/71.97**	64.91/57.73	68.23/**69.46**	60.98/**68.17**	72.40/64.87	64.12/66.86

jective may reinforce incorrect decision tendencies and lead to imbalanced cover–stego detection. In contrast, our method keeps the source-trained model frozen and only edits the intermediate representation of each test sample along the source-derived cover–stego direction. This makes the editing process lightweight and less sensitive to noisy target predictions.

However, the improvement is not uniform across all steganographic algorithms. For iMEC, our method does not outperform the BERT-FT baseline on average, even with the selected algorithm-level coefficient. A possible reason is that iMEC places stronger emphasis on distributional security, aiming to make the generated stego texts distributionally close to cover texts. As a result, the cover–stego discrepancy induced by iMEC is usually weaker and more easily affected by corpus-specific topics, vocabularies, and generation patterns. In this case, the cover–stego direction estimated from the source domain may not mainly reflect a stable and transferable steganographic direction, but instead contain more source-specific discriminative cues. When this direction is applied to iMEC texts from unseen target domains, the projection score along this direction may fail to accurately indicate the strength of steganographic evidence, thereby limiting the effectiveness of representation editing along a single source-derived direction.

5 Conclusion

In cross-domain linguistic steganalysis, a detector trained on source-domain data often suffers from a mismatch between shifted target-domain representations and the decision boundary learned from the source domain. To address this problem, we proposed REED, a post-training representation editing framework. REED first trains a steganalysis classifier on labeled source-domain data and then keeps the feature extractor and classifier fixed. Instead of adapting the model by modifying parameters, adding training objectives, or moving the learned decision boundary, REED directly edits the intermediate representations of test samples before classification. This framework is instantiated under both domain adaptation and domain generalization settings: in domain adaptation, a source–target domain-offset vector is used to compensate for cross-domain shifts; in domain generalization, a source-derived cover–stego direction guides sample-specific editing for unseen-domain texts. Experiments on multiple corpora and steganographic algorithms show that REED improves cross-domain detection performance in most settings, especially in terms of F1-score, while requiring no architecture modification or parameter updates after source-domain training. These results indicate that feature-space representation editing provides a lightweight alternative to parameter-update-based adaptation. In future work, we will explore more robust vector estimation and automatic coefficient selection to further improve the adaptability of REED under more complex cross-domain scenarios.

6 Limitation

Although REED improves performance on most cross-domain linguistic steganalysis tasks, it still mainly relies on mean representations to construct editing vectors. This design is simple and efficient, but a single global vector may not fully capture fine-grained differences under complex distributions. When steganographic traces in the target domain differ substantially from those in the source domain, or are highly entangled with domain-specific topics and writing styles, the effectiveness of representation editing may be limited. The results on iMEC also show that the source-derived cover–stego direction does not always transfer stably to all target domains.

In addition, the current domain generalization experiments still require selecting the editing coefficient λ at the steganographic-algorithm level. Although the same coefficient is fixed across all transfer directions for each algorithm, how to automatically determine an appropriate editing strength under a fully target-free setting remains an open problem. In future work, we will explore more robust vector estimation, adaptive direction selection, and automatic coefficient selection, and further evaluate REED on more text domains, steganographic algorithms, and pre-trained models.

References

Ross J. Anderson and Fabien A. P. Petitcolas. 1998. On the limits of steganography. *IEEE Journal on Selected Areas in Communications*, 16(4):474–481.

Qi Cao, Xingming Sun, and Lingyun Xiang. 2013. A Secure Text Steganography Based on Synonym Substitution. In *IEEE Conference Anthology*, pages 1–3. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tina Fang, Martin Jaggi, and Katerina Argyraki. 2017. Generating Steganographic Text with LSTMs. In *Proceedings of ACL 2017, Student Research Workshop*, pages 100–106, Vancouver, Canada. Association for Computational Linguistics.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter Sentiment Classification Using Distant Supervision. CS224N Project Report, Stanford University.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A Kernel Two-Sample Test. *Journal of Machine Learning Research*, 13(25):723–773.

Gabriel Kaptchuk, Tushar M. Jois, Matthew Green, and Aviel D. Rubin. 2021. Meteor: Cryptographically secure steganography for realistic distributions. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 1529–1548. ACM.

Yufei Luo, Zhen Yang, Ru Zhang, and Jianyi Liu. 2025. Pseudo-label Based Domain Adaptation for Zero-Shot Text Steganalysis. In *Computational and Experimental Simulations in Engineering*, volume 176 of *Mechanisms and Machine Science*, pages 128–142, Cham. Springer.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Wojciech Mazurczyk and Steffen Wendzel. 2018. Information hiding: Challenges for forensic experts. *Communications of the ACM*, 61(1):86–94.

Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. 2022. Efficient Test-Time Model Adaptation Without Forgetting. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16888–16905. PMLR.

Yan Niu, Juan Wen, Ping Zhong, and Yiming Xue. 2019. A hybrid R-BILSTM-C neural network based text steganalysis. *IEEE Signal Processing Letters*, 26(12):1907–1911.

Wanli Peng, Sheng Li, Zhenxing Qian, and Xinpeng Zhang. 2023. Text Steganalysis Based on Hierarchical Supervised Learning and Dual Attention Mechanism. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:3513–3526.

Wanli Peng, Jinyu Zhang, Yiming Xue, and Zhenghong Yang. 2021. Real-Time Text Steganalysis Based on Multi-Stage Transfer Learning. *IEEE Signal Processing Letters*, 28:1510–1514.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Christian Schroeder de Witt, Samuel Sokota, J. Zico Kolter, Jakob Nicolaus Foerster, and Martin Strohmeier. 2023. Perfectly Secure Steganography Using Minimum Entropy Coupling. In *International Conference on Learning Representations*.

Mohammad Shirali-Shahreza. 2008. Text Steganography by Changing Words Spelling. In *2008 10th International Conference on Advanced Communication Technology*, volume 3, pages 1912–1913. IEEE.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. 2020. Test-Time Training With Self-Supervision for Generalization Under Distribution Shifts. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9229–9248. PMLR.

Andrew Thompson. 2017. All the News: 143,000 Articles from 15 American Publications. Kaggle Dataset, Version 4.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. 2021. Tent: Fully Test-Time Adaptation by Entropy Minimization. In *International Conference on Learning Representations*.

Juan Wen, Xuejing Zhou, Ping Zhong, and Yiming Xue. 2019. Convolutional neural network based text steganalysis. *IEEE Signal Processing Letters*, 26(3):460–464.

Hanzhou Wu, Biao Yi, Feng Ding, Guorui Feng, and Xinpeng Zhang. 2021. Linguistic Steganalysis With Graph Neural Networks. *IEEE Signal Processing Letters*, 28:558–562.

Jiaxuan Wu, Xin Chen, Juan Wen, Wanli Peng, and Yiming Xue. 2024. A Test-Time Entropy Minimization Method for Cross-Domain Linguistic Steganalysis. *IEEE Signal Processing Letters*, 31:1760–1764.

Lingyun Xiang, Xingming Sun, Gang Luo, and Bin Xia. 2014. Linguistic Steganalysis Using the Features Derived from Synonym Frequency. *Multimedia Tools and Applications*, 71:1893–1911.

Yiming Xue, Lingzhi Kong, Wanli Peng, Ping Zhong, and Juan Wen. 2022a. An Effective Linguistic Steganalysis Framework Based on Hierarchical Mutual Learning. *Information Sciences*, 586:140–154.

Yiming Xue, Jiaxuan Wu, Ronghua Ji, Ping Zhong, Juan Wen, and Wanli Peng. 2024. Adaptive Domain-Invariant Feature Extraction for Cross-Domain Linguistic Steganalysis. *IEEE Transactions on Information Forensics and Security*, 19:920–933.

Yiming Xue, Boya Yang, Yaqian Deng, Wanli Peng, and Juan Wen. 2022b. Domain Adaptational Text Steganalysis Based on Transductive Learning. In *Proceedings of the 2022 ACM Workshop on Information Hiding and Multimedia Security*, pages 91–96. ACM.

Jinshuai Yang, Zhongliang Yang, Jiajun Zou, Haoqin Tu, and Yongfeng Huang. 2023. Linguistic Steganalysis Toward Social Network. *IEEE Transactions on Information Forensics and Security*, 18:859–871.

Zhen Yang, Yufei Luo, Jinshuai Yang, Xin Xu, Ru Zhang, and Yongfeng Huang. 2025. Class-Aware Adversarial Unsupervised Domain Adaptation for Linguistic Steganalysis. *IEEE Transactions on Information Forensics and Security*, 20:5181–5194.

Zhong-Liang Yang, Xiao-Qing Guo, Zi-Ming Chen, Yong-Feng Huang, and Yu-Jin Zhang. 2019a. RNN-Stega: Linguistic steganography based on recurrent neural networks. *IEEE Transactions on Information Forensics and Security*, 14(5):1280–1295.

Zhongliang Yang, Shuyu Jin, Yongfeng Huang, Yujin Zhang, and Hui Li. 2018. Automatically Generate Steganographic Text Based on Markov Model and Huffman Coding. *Preprint*, arXiv:1811.04720.

Zhongliang Yang, Ke Wang, Jian Li, Yongfeng Huang, and Yu-Jin Zhang. 2019b. TS-RNN: Text steganalysis based on recurrent neural networks. *IEEE Signal Processing Letters*, 26(12):1743–1747.

Siyu Zhang, Zhongliang Yang, Jinshuai Yang, and Yongfeng Huang. 2021. Provably secure generative linguistic steganography. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3046–3055, Online. Association for Computational Linguistics.

Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. 2023. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4396–4415.

Xuejing Zhou, Wanli Peng, Boya Yang, Juan Wen, Yiming Xue, and Ping Zhong. 2022. Linguistic steganography based on adaptive probability distribution. *IEEE Transactions on Dependable and Secure Computing*, 19(5):2982–2997.

Zachary M. Ziegler, Yuntian Deng, and Alexander M. Rush. 2019. Neural linguistic steganography. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1210–1215, Hong Kong, China. Association for Computational Linguistics.

A Editing Coefficients

Table 4 reports the algorithm-level editing coefficients selected from the predefined search range in the Domain Generalization experiments. Specifically, we search λ in the range of $[0, 10]$ with a step size of 0.1. For each steganographic algorithm, the same coefficient is used for all six source–target transfer directions. The coefficient λ controls the strength of the steganographic direction editing.

Algorithm	AC	HC	ADG	Meteor	iMEC
λ	5.0	2.4	0.4	0.2	0.1

Table 4: Algorithm-level editing coefficients selected from the predefined search range in the Domain Generalization experiments.