Rewriting-Stego: Generating Natural and Controllable Steganographic Text with Pre-trained Language Model

Fanxiao Li[1], Sixing Wu[2(✉)], Jiong Yu[1], Shuoxin Wang[1], BingBing Song[3], Renyang Liu[3], Haoseng Lai[1], and Wei Zhou[2]

[1] Engineering Research Center of Cyberspace, Yunnan University, Yunnan, China
lifanxiao@mail.ynu.edu.cn
[2] National Pilot School of Software, Yunnan University, Yunnan, China
{wusixing,zwei}@ynu.edu.cn
[3] School of Information Science and Engineering, Yunnan University, Yunnan, China

Abstract. Data transmission security and privacy play a crucial role in the era of information technology. Although the widely-used data encryption technique can ensure security, it can be easily detected and blocked by the observation system because the encrypted data format is quite different from the normal data. This work focuses on linguistic steganography, hiding a secret text in another normal stego text to ensure security and decrease the risk of being detected simultaneously. Rather than following the existing edit-based or generation-based paradigm, we propose a novel rewriting-based *Rewriting-Stego*, which tries to hide a secret text in the stego text by rewriting the given cover text. This paradigm integrates the advantages of both the edit-based paradigm and the generation-based paradigm, bringing higher information capacity without losing naturalness and controllability. Extensive experimental results on three public datasets have demonstrated the effectiveness of our *Rewriting-Stego* in terms of multiple metrics.

Keywords: steganography · linguistic steganography

1 Introduction

The Internet's rapid development arouses concerns about security and privacy because unauthorized attackers can easily intercept the transmitted data in non-dedicated networks. As shown in Fig. 1, data encryption is the most widely used security technique. The *sender* first uses a key to encrypt the data; then, the ciphertext can be transmitted via the Internet, and only the *receiver* who has another key can correctly decrypt the ciphertext. Nonetheless, the data format of ciphertext is quite different from the normal data, which may cause the vigilance of the observation system [1], and the data transmission may be blocked. Unlike data encryption, data steganography hides the secret message in a stego message and keeps a normal data format. Thus, data steganography

X. Wang et al. (Eds.): DASFAA 2023, LNCS 13943, pp. 617–626, 2023.
https://doi.org/10.1007/978-3-031-30637-2_41

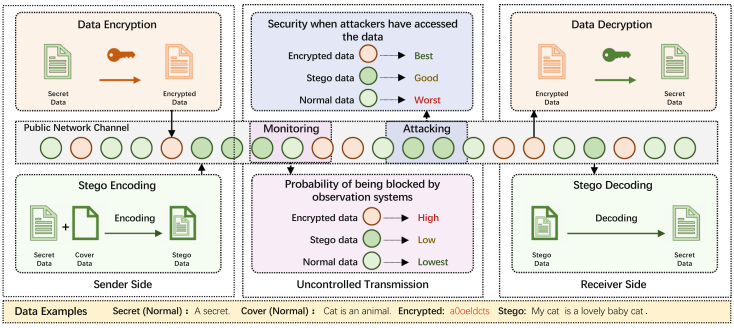

Fig. 1. The comparison between data encryption and data steganography

can reduce the vigilance from the observation system and has received much attention in security communication [2], watermarking [5,16], etc.

This paper studies linguistic steganography [4,13,17], which hides a secret text in the stego (i.e., *steganographic*) text given a cover text. Roughly, prior works are either the *edit-based* [14] or the *generation-based* [4,15]. Edit-based methods design a special encoding strategy to hide the secret text in some selected positions of the cover text via editing. For example, given a synonym dictionary, replacing a word with the 3rd ranked synonym can hide 2-bit[1] information 11. However, to ensure the naturalness of the stego text, the information capacity of the carried secret message is always limited. The average BPT (bit per token) is always less than 1.0. With the development of language models (LM) [11,12], generation-based methods have become mainstream. Generation-based methods first use a cover text to initialize the state of the backbone LM; subsequently, the LM generates a sequel text as the stego text based on the given secret bit stream and the decoding strategy. For example, at each generation step, the LM first outputs a probability distribution of the next token; then, rather than selecting the most possible token or randomly sampling a token, the strategy assigns bit encoding codes to candidate tokens according to the rank of the corresponding probability and selects the token whose bit code equals to the current secret bit code. This paradigm can achieve a higher information capacity (>1 BPT). However, the generated stego text 1) always lacks naturalness and 2) is hard to control the content because the supervision is limited during the generation process, increasing the chance to raise the vigilance of the observation system.

With such challenges in mind, we propose a novel *rewriting-based* method *Rewriting-Stego*, which rewrites the cover text and lets the outputted paraphrased text as the stego text. We regard the rewriting as a denoising sequence-to-sequence task; namely, given an input text, the model should denoise the unwanted information and then generate an output text with the same semantics but different word usage. In our context, the cover text is the input, the

[1] Generally, the stego text is represented as a bit stream.

stego text is the output, and the secret text serves as a restriction to denoise. Consequently, we choose a pre-trained sequence-to-sequence BART [8] as our backbone language model. We propose a Plug-and-Play Group-Wise Masked Decoding Strategy to hide the secret message without affecting the structure of the backbone BART. We group the vocabulary of the BART into 2^n groups; each group has a unique n-bit code, and each token only belongs to a group. Thus, the backbone BART first encodes the given cover text; then, in the decoding stage, we can hide n-bit secret information in each generated token by masking tokens whose group bit-id is not equal to the current secret information. Then, we propose to use a text-based Condition Codes to explicitly hint the desired length of the stego text in the encoding stage and a beam-search-based *Beam-then-Rank* to select higher-quality stego text in the decoding stage. Finally, we propose Adaptive Fine-Tuning to help the backbone adjust to the rewriting-based linguistic steganography task. Intuitively, the proposed rewriting-based paradigm combines the advantages of the previous two paradigms. On the one hand, similar to the edit-based methods [14], the generated stego text is highly similar to the cover text, bringing higher naturalness and controllability. On the other hand, similar to the generation-based methods [4,13,15], the rewriting-based method can reach higher information capacity because all tokens in the generated stego text can hide secret information. Our code is available at https://github.com/cheslee15/Rewriting-Stego.

2 Methodology

2.1 Problem Definition

Linguistic steganography task can be formulated as the *sender* hides a secret message S into a cover text Y (a text about other normal topics) and obtains a stego text Y' via the pre-defined invertible strategy. The stego text Y' is very similar to the original cover text Y because they describe a similar topic and use a similar format on the surface text. Thus, the *observation system* can hardly detect the anomaly that exists in the stego text. Unlike the observation system, the *reciever* can restore the secret message S from the stego text Y' via the pre-defined invertible strategy.

2.2 Rewriting Paradigm

Rather than following the generation-based or edit-based paradigm, this paper proposes a novel **rewriting-based** paradigm. Similar to the generation-based paradigm, the proposed rewriting-based paradigm employs a language model (LM) to generate the stego text. However, rather than generating a sequel text to hide S, our rewriting-based paradigm rewrites the cover text and regards the obtained *paraphrased text* as the stego text. By definition, a paraphrased text $Y^{'}$ has the same semantics as the original cover text Y but different word usage.

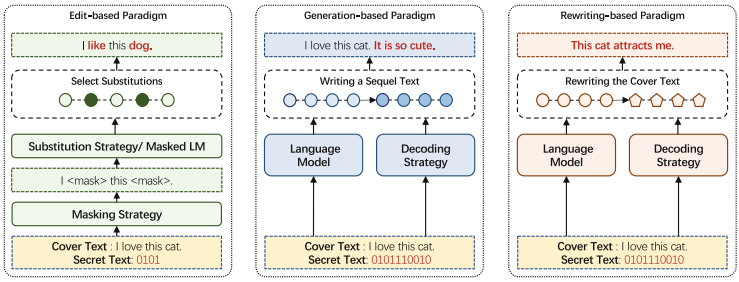

Fig. 2. The comparison among three different linguistic steganography paradigms.

Thus, similar to the edit-based paradigm, the generation process of the paraphrased text is highly supervised by the original cover text, bringing higher naturalness and controllability. In addition, compared to the edit-based paradigm, our rewriting-based paradigm can edit all tokens in the cover text, bringing higher information capacity without losing naturalness and controllability. Consequently, the essentials of a rewriting-based method are 1) a rewriting language model and 2) a strategy to hide the secret text S.

2.3 Methodology

Backbone Model. We employ BART [8], a denoising auto-encoder for Seq2Seq tasks, as our backbone language model. In the pre-training stage, given an original text $X_{original}$, a corrupted $X_{corrupt}$ is subsequently synthesized by adding the manually defined noises to the original text $X_{original}$. Then, the objective of BART is to restore $X_{original}$ given the corrupted $X_{corrupt}$. Consequently, BART is very suitable for the rewriting-based paradigm because it has the ability to denoise the input text and generate a higher-quality paraphrased text.

Encoding with the Conditional Code. In linguistic steganography, the length of the stego text $Y^{'}$ depends on the length of the secret message S, rather than the length of the inputted cover text Y. If the generated stego text is incomplete or strange, it may increase the risk of being blocked. To alleviate this issue, we propose to encode the cover text with a *Conditional Code*, which involves an explicit length signal from the secret text S. Given a cover text $Y = (y_1, y_2, \cdots, y_n)$, *Rewriting-Stego* first employs the BART encoder to encode and obtain $\mathbf{H} = Encoder([Y; ConditionalCode])$ where the input is the concatenation of the cover text Y and a conditional code $ConditionalCode$. Previous studies [7] have shown the potential of promoting text in the pre-trained language model. Inspired by this, our $ConditionalCode$ uses a promoting text *'Generate a sentence of length L by paraphrasing the content on the left.'* to explicitly indicate the length of $Y^{'}$ should be L.

Plug-and-Play Group-Wise Masked Decoding Strategy. Subsequently, the decoder of BART continues to generate the stego text $Y^{'}$. To hide the secret message S into the generated stego text $Y^{'}$, we propose a *Plug-and-Play Group-Wise Masked Decoding Strategy*. This strategy neither modify the network structure nor restrict the selection of the decoding algorithm.

Similar to previous works, the secret message S should be encoded as a bit stream (i.e., a sequence of bits), and the total bit length of the secret message S is denoted as l. Then, we assume each token $\in Y^{'}$ hides n bits secret message. Subsequently, in each generation time step t, we generate a stego token $y_t^{'}$ to hide the current secret message bits $S_{(t-1)*n:t*n}$:

$$\begin{aligned} y_t^{'} &= DecodingStrategy(S_{(t-1)*n:t*n}, P(y_t^{'}|Y_{1:t}^{'}; Y; ConditionalCode)) \\ P(y_t^{'}|Y_{1:t}^{'}; Y; ConditionalCode) &= Softmax(MLP(Decoder(Y_{1:t}^{'}, \mathbf{H}))) \end{aligned} \tag{1}$$

where $P(y_t^{'}|Y_{1:t}^{'}; Y; ConditionalCode)$ is the current token prediction probability distribution over the vocabulary, which is outputted by the BART decoder; the vocab predictor MLP is a feed-forward neural network. The generation process is restricted by the current secret bits $S_{(t-1)*n:t*n}$ and the *Decoding Strategy*. As illustrated in Table 1, *Rewriting-Stego* divides the vocabulary into 2^n groups and assigns an n-bits group bit id. If the vocabulary has $|V|$ tokens in total, then each group has an n-bit id and $\frac{|V|}{2^n}$ tokens. Thus, each token in the vocabulary corresponds to a deterministic n-bit code.

Table 1. Vocabulary-based grouping strategy (Modulo Operation).

Group Bit ID	Tokens	Group Bit ID	Tokens
00	{“be”:0, “it”:4...}	01	{“this”:1, “who”:5...}
10	{“he”:2, “an”:6...}	11	{“from”:3, “much”:7...}

Subsequently, the current secret message bits $S_{(t-1)*n:t*n}$ can be uniquely aligned to one vocabulary group G_t. Then, *Decoding Strategy* will mask a token probability to zero if this token is excluded by the aligned group G_t. For example, if the current secret message bits are 01, we mask the token probabilities in the other groups (00,10,11) to 0. After masking the invalid tokens, the following generation process is the same as the original BART.

Beam-then-Rank. *Rewriting-Stego* can freely select greedy search, beam search, or any other common algorithm to select the final prediction of $Y_t^{'}$. Thus, to improve the generation quality, we design a *Beam-then-Rank*: 1) we first use beam search to generate K stego candidates; 2) then, we use an external GPT2 model to estimate the PPL and then select the best candidate.

Restoring. The receiver who knows the vocabulary grouping can restore the secret message S from the stego text $Y^{'}$ by checking the group bit id.

Fine-Tuning. We believe conducting fine-tuning can deliver better performance. Thus, we synthesize a fine-tuning dataset via the data augmentation technique. We first sample one million high-quality instances F whose perplexity (PPL) is greater than 20 and less than 200. Then, we employ the augmentation tool [9] to synthesize a parallel perturbed dataset, which includes 8 word-level perturbation operations: 1) random insertion, 2) random substitution, 3) synonym substitution, 4) antonym substitution, 5) word decomposition, 6) deletion, 7) transposition, and 8) random combination of the preceding methods. Finally, we mix the augmented data with the original data and randomly select one million data as input and the original text of these data as labels to form the fine-tuning dataset. In the fine-tuning process, we randomly masked about 75% of the input. We adopt AdamW as the optimizer; the batch size is set to 512; the learning rate is set to 3.5e−5, and the warm-up strategy is used with a warm-up number of 8000 steps. Finally, to avoid over-fitting, we only fine-tune 1 epoch.

3 Experiments

3.1 Settings

We evaluate models on three public datasets, namely, *Large Movie Review Dataset (**Movie**)* [10], *All the News (**News**)*, *Sentiment140 (**Tweet**)* [6]. For all datasets, raw texts are first converted to lowercase, and HTML tags and most punctuations are removed. All texts are tokenized by the NLTK tools, and then sentences whose length is below 5 or above 200 are filtered. Next, several methods are selected as baselines, where **Bins** [4], **Huffman** [15], and **Saac** [13]are generation-based methods, **Masked-Stega** [14] is an edit-based method. The first three generation-based baseline methods are implemented through the source code released by Saac [13]. Masked-Stega is implemented through the official source code, and we set p to 0.01. For Bins, we set b to be 4, and the corresponding number of bins is 16. For Huffman, we build the Huffman tree with the top 128 likely tokens. For Saac, we chose the imperceptibility gap δ to be 0.01. For our method, we use *bart-base*[2] as the backbone, and the beam width is set to be 50. For all models, we have sampled 5000 texts as the cover texts and another 5000 texts as the secret messages, and use the following metrics: **1) Bits per Token (BPT)** measures the information capacity of stego text, it reports the average number of hided secret bits per token(word) in the generated stego text. A larger BPT indicates that the method can carry more secret information in the same-length stego text; **2) Perplexity (PPL)** is a language modeling metric that measures the quality of the given text from the perspective of probability. A smaller PPL means that the generated sentences are more natural.

[2] BART, 140M, https://huggingface.co/facebook/bart-base.

Here, we use a pre-trained *gpt2-medium* as the backbone language model; **3) Mean and Variance** measures the imperceptibility (anti-steganalysis ability). For a stego text, we mask each token to [MASK] in turn and use BERT [3] to get the sorted word predictions at the masked position. Then we collect the position of each original token in the sorted predictions. Finally, we calculate the mean and the variance of positions of the original words. A smaller mean and variance indicate higher imperceptibility, and the generated stego text is easier to avoid the detection of the observation system; **4) Detection Accuracy (ACC)** also measures the imperceptibility. We fine-tuned a BERT as a classifier to detect whether the text is stego text. We sampled 30,000 texts from the three mentioned datasets as the normal texts and generated 30,000 stego texts using the Arithmetic Coding [17].

3.2 General Results

Table 2. Evaluation Results. BPT_{C+S} considers the transmission of the cover text if required, but BPT_S does not. *: the secret message can not be entirely encoded.

Dataset	Method	BPT_S	BPT_{C+S}	PPL	Mean	Variance	Acc
Movie	Vanilla	0.0	0.0	114.9	162.4	2.7e04	1.6%
	Bins	4.0	4.0	332.2	170.4	1.2e05	67.6%
	Huffman	4.8	0.79	284.6	224.6	3.4e05	71.5%
	Saac	**5.3**	0.98	635.4	319.3	2.9e05	72.9%
	Masked-Stega*	0.16	0.16	126.4	148.5	2.5e04	**7.1%**
	Rewriting-Stego	1.0	1.0	105.2	121.4	**2.1e04**	13.7%
	Rewriting-Stego	2.0	2.0	**72.5**	**71.8**	2.7e04	20.3%
	Rewriting-Stego	4.0	**4.0**	130.1	87.1	6.6e04	23.4%
News	Vanilla	0.0	0.0	92.7	175.5	2.8e04	2.1%
	Bins	4.0	4.0	424.5	237.2	2.4e05	59.6%
	Huffman	4.7	0.76	346.7	269.9	3.8e05	75.5%
	Saac	**5.1**	0.93	586.9	327.5	2.9e05	78.1%
	Masked-Stega*	0.25	0.25	119.8	159.2	**2.5e04**	**2.2%**
	Rewriting-Stego	1.0	1.0	114.2	152.9	2.6e04	8.9%
	Rewriting-Stego	2.0	2.0	**101.6**	**119.3**	4.2e04	18.9%
	Rewriting-Stego	4.0	**4.0**	158.1	124.7	1.4e05	20.6%
Tweet	Vanilla	0.0	0.0	183.5	180.0	5.1e04	13.9%
	Bins	4.0	4.0	908.8	278.2	3.3e05	46.4%
	Huffman	4.7	0.87	1223.9	366.7	7.6e05	40.6%
	Saac	**5.4**	1.16	1924.3	369.9	6.0e05	51.8%
	Masked-Stega*	0.16	0.16	196.4	158.3	4.5e04	31.1%
	Rewriting-Stego	1.0	1.0	69.7	70.1	**2.1e04**	**18.4**%
	Rewriting-Stego	2.0	2.0	**62.1**	**44.4**	2.2e04	19.2%
	Rewriting-Stego	4.0	**4.0**	137.3	109.3	1.4e05	42.1%

Table 2 reports the result. It must be noted that Masked-Stega can not entirely hide the secret message in most cases because it strictly requires the length of the

cover text is linearly related to the length of the secret message (about 3X-5X longer than the secret message).

Comparison with Generation-based: When BPT_S is set to 4, *Rewriting-Stego* can outperform generation-based baselines with similar information capacity. *Rewriting-Stego* have lower scores in Mean, Variance, and Acc, indicating the generated stego text will raise less attention from the observation system. *Rewriting-Stego* also has a significantly lower PPL. It shows *Rewriting-Stego* can generate more natural stego text. Finally, generation-based models need the cover text to initialize the backbone language model when restoring the secret message; thus, we have to consider the transmission of the cover text at the same time. *Rewriting-Stego* does not have this issue, bringing higher real-world information capacity in terms of BPT_{C+S}. **Comparison with Edit-based:** The edit-based Masked-Stega has significantly lower information capacity than others. If we set the BPT_S to 1.0, *Rewriting-Stego* has at least 3–4 times higher information capacity, and the overall performance is still better than Masked-Stega in most comparisons. The major advantage of Masked-Stega is the lower detection accuracy (Acc), which may be better than *Rewriting-Stego* in some specific anti-steganalysis systems. However, *Rewriting-Stego* has better performance in almost other metrics. **Comparison with the Vanilla:** We also evaluate the human-generated cover text in the same way. Besides the detection accuracy, our *Rewriting-Stego* has better performance. Such results are not strange because 1) We find such human-generated cover texts have various noises; 2) The backbone model of *Rewriting-Stego* is BART, which is also a denoising model. Thus, our *Rewriting-Stego* can denoise the input along with generating the stego text.

3.3 More Studies

Ablation Study. We conduct experiments to analyze the impact of the following terms: 1) Fine-Tune, 2) Conditional Code, and 3) Beam-then-Rank. The first *base* model uses the pre-trained BART with no advanced technique. Afterward, we gradually add the Fine-Tine (*base+FT*), Conditional Code (*base+FT+CC*), and Beam-then-Rank (*base+FT+BR*). As reported in Fig. 3: 1) The naive *base* still has a competitive performance compared to baselines (see Table 2), showing the notable advantage of rewriting-paradigm; 2) After the fine-tuning, the scores have notable improvements in all metrics. It shows this procedure will help *Rewriting-Stego* to generate higher-quality stego text; 3) The Conditional Code mainly helps *Rewriting-Stego* reduce the Mean and the Variance; 4) Beam-than-Rank has also notably improved the performance. This is why we design a plug-and-play decoding strategy to be compatible with most decoding algorithms.

Case Study. We have sampled two examples in Table 3. It can be observed that: 1) When BPT_S=1, the stego text generated by *Rewriting-Stego* is highly similar to the cover text, which verified the superior naturalness and controllability of our approach; 2) With the increasing of BPT_S, the generated stego texts become

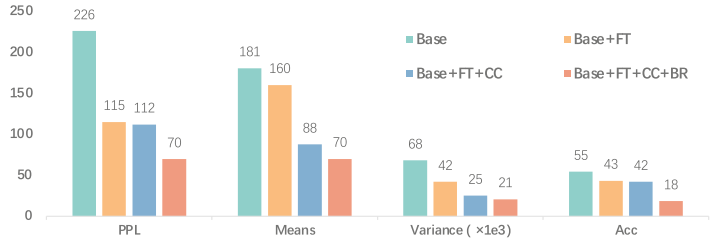

Fig. 3. Ablation Study on the Tweet Dataset. BPT_S is set to 1.0 in the experiments.

Table 3. Examples of stego texts. Limited by the space, we omitted the padding part.

24-bit Secret Message : (1,0,0,0,1,0,0,1,0,1,1,0,1,0,1,0,1,1,1,1,1,0,1,1)		
Cover Text	Method	Stego Text
sometimes we bring the story to you, sometimes you have to go to the story.	Rewriting-Stego BPT_S=1	sometimes we bring it to your mind but sometimes you have to go back to it to find the story you want to tell...
	Rewriting-Stego BPT_S=2	but we have a different way of thinking about what the story...
	Rewriting-Stego BPT_S=4	some of all the story lines...
	Bins BPT_S=4	somewhere and we have lots different...
	Masked-Stega (BPT_S=0.16)(Encrypted message :10001)	sometimes we read the story to you , sometimes you have to stick to the story...

shorter, and the similarity to the cover text is weaker; 3) The generation-based Bins and our *Rewriting-Stego* can entirely hide the secret message but need the padding operation to make the stego text complete; the edit-based Masked-Stega can generate a complete stego text but can not entirely hide the secret message.

4 Conclusion

This paper proposes *Rewriting-Stego*, a novel rewriting-based linguistic steganographic approach. *Rewriting-Stego* aims to improve the information capacity of the stego text without losing the naturalness and controllability. We use a pre-trained BART as the backbone l model and propose a *Plug-and-Play Group-Wise Msked Decoding Strategy* to rewrite the given cover text and hide the secret message in the obtained paraphrased text (stego text). Besides, *Rewriting-Stego* uses a text-based *Condition Code* and *Beam-then-Rank* strategy to deliver better performance. Experimental results show that *Rewriting-Stego* outperforms the baselines in most metrics.

Acknowledgment. This work was supported in part by the National Natural Science Foundation of China under Grant 62162067 and 62101480, in part by the Yunnan Province Science Foundation under Grant No. 202005AC160007, No. 202001BB050076, and Research and Application of Object detection based on Artificial Intelligence, in part by the Applied Basic Research Foundation of Yunnan Province under Grant 202201AT070156, in part by the Fund project of Yunnan Province Education Department "Generating mnatural and controllable steganographic text based on language model".

References

1. Bernaille, L., Teixeira, R.: Early recognition of encrypted applications. In: Uhlig, S., Papagiannaki, K., Bonaventure, O. (eds.) PAM 2007. LNCS, vol. 4427, pp. 165–175. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-71617-4_17
2. Bi, X., Yang, X., Wang, C., Liu, J.: High-capacity image steganography algorithm based on image style transfer. Secur. Commun. Netw. **2021** (2021)
3. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
4. Fang, T., Jaggi, M., Argyraki, K.J.: Generating steganographic text with LSTMs. In: ACL (2017)
5. Garg, M., Gupta, S., Khatri, P.: Fingerprint watermarking and steganography for ATM transaction using LSB-RSA and 3-DWT algorithm. In: ICCN, pp. 246–251 (2015)
6. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. CS224N project report, Stanford **1**(12) (2009)
7. Lester, B., Al-Rfou, R., Constant, N.: The power of scale for parameter-efficient prompt tuning. arXiv preprint arXiv:2104.08691 (2021)
8. Lewis, M., et al.: BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: ACL (2020)
9. Ma, E.: Nlp augmentation (2019)
10. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: ACL (2011)
11. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: Interspeech, vol. 2 (2010)
12. Qiu, X.P., Sun, T.X., Xu, Y.G., Shao, Y.F., Dai, N., Huang, X.J.: Pre-trained models for natural language processing: a survey. Sci. China Technol. Sci. **63**(10), 1872–1897 (2020). https://doi.org/10.1007/s11431-020-1647-3
13. Shen, J., Ji, H., Han, J.: Near-imperceptible neural linguistic steganography via self-adjusting arithmetic coding. In: EMNLP (2020)
14. Ueoka, H., Murawaki, Y., Kurohashi, S.: Frustratingly easy edit-based linguistic steganography with a masked language model. In: NAACL (2021)
15. Yang, Z., Guo, X., Chen, Z., Huang, Y., Zhang, Y.: RNN-stega: linguistic steganography based on recurrent neural networks. IEEE Trans. Inf. Forensics Secur. **14**(5) (2019)
16. Zhang, C., Benz, P., Karjauv, A., Sun, G., Kweon, I.S.: UDH: universal deep hiding for steganography, watermarking, and light field messaging. In: NeurIPS (2020)
17. Ziegler, Z.M., Deng, Y., Rush, A.M.: Neural linguistic steganography. In: EMNLP-IJCNLP (2019)