

# 言語ステガノグラフィ

京都大学  
村脇 有吾



共同研究者: 上岡 輔乃 (当時B4)

野崎 樹文 (当時M1学内インターン)

巖 睿逸 (現博士後期課程)

[\[NAACL 2021\]](#)

[\[AACL 2022\]](#)

[\[EMNLP 2025\]](#) [\[preprint\]](#)

# 言語ステガノグラフィとは？

夕方に川沿いを歩いた。風が冷たくて、手袋が欲しくなる。遠くの山がきれいに見えて、空が高い。帰りにコンビニで温かい飲み物を買った。こういう日が続くといい。

# 言語ステガノグラフィとは？

秘密のメッセージ

実験は失敗していたのに「成功」にされた。生データは見せてもらえず、図だけが先にできていた。私は改ざん前のログを保存している。公開される前に止めるべきだ。

埋め込み

抽出

ステゴテキスト

夕方に川沿いを歩いた。風が冷たくて、手袋が欲しくなる。遠くの山がきれいに見えて、空が高い。帰りにコンビニで温かい飲み物を買った。こういう日が続くといい。

# 言語ステガノグラフィとは？

秘密のメッセージ

実験は失敗していたのに「成功」にされた。生データは見せてもらえず、図だけが先にできていた。私は改ざん前のログを保存している。公開される前に止めるべきだ。

符号化

復号化

秘密のメッセージ  
(ビット列)

```
011011100001101110100000100111010011011111010000101110100111110001101100
000100010011001100101100011100010110011110001011111111110101100110101100
111010100011110011011011011010110000011000010111100110111000101110011011000
0010010110000011101110111010111010010011010011111001101110111010
```

埋め込み

抽出

ステゴテキスト

夕方に川沿いを歩いた。風が冷たくて、手袋が欲しくなる。遠くの山がきれいに見えて、空が高い。帰りにコンビニで温かい飲み物を買った。こういう日が続くといい。

# 言語ステガノグラフィとは？

秘密のメッセージ

実験は失敗していたのに「成功」にされた。生データは見せてもらえず、図だけが先にできていた。私は改ざん前のログを保存している。公開される前に止めるべきだ。

符号化

復号化

秘密のメッセージ  
(ビット列)

```
011011100001101110100000100111010011011111010000101110100111110001101100
000100010011001100101100011100010110011110001011111111110101100110101100
111010100011110011011011010110000011000010111100110111000101110011011000
0010010110000011101110111010111010010011010011111001101110111010
```

埋め込み

抽出

ステゴテキスト

夕方に川沿いを歩いた。風が冷たくて、手袋が欲しくなる。遠くの山がきれいに見えて、空が高い。帰りにコンビニで温かい飲み物を買った。こういう日が続くといい。

# 言語ステガノグラフィ：タスク要件

秘密のメッセージ  
(ビット列)

01101110000110111010000010011101001101111101000010111010011110001101100000100010  
0110011001011000111000101100111000101111111101011001101100111010100011110011  
0110110110110000110000101110011011100010111001101100000100101100000111011101  
011101001001101001111100110111011010

埋め  
込み

ステゴテキスト

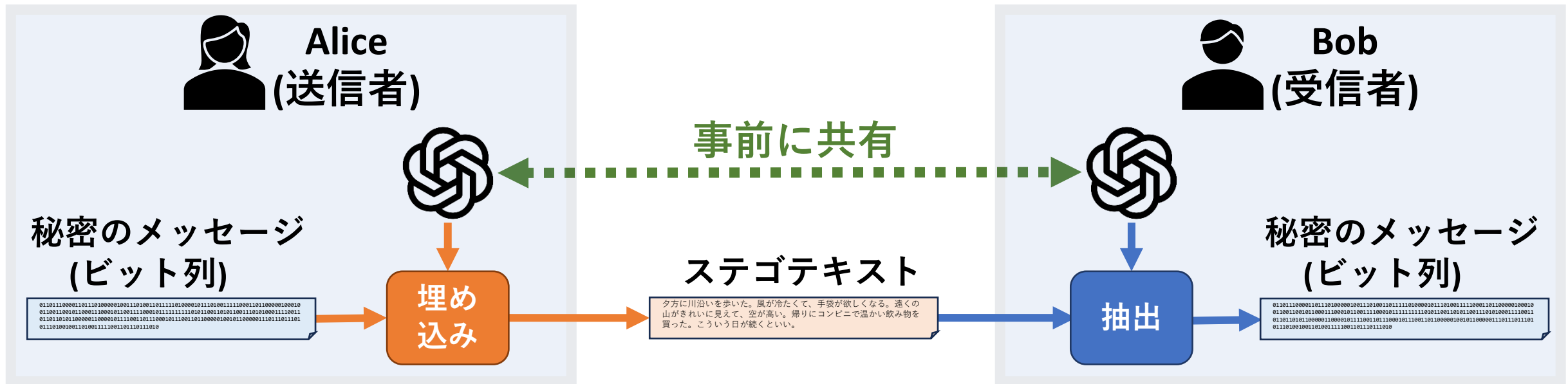
夕方に川沿いを歩いた。風が冷たくて、手袋が欲しくなる。遠くの山がきれいに見えて、空が高い。帰りにコンビニで温かい飲み物を買った。こういう日が続くといい。

抽出

秘密のメッセージ  
(ビット列)

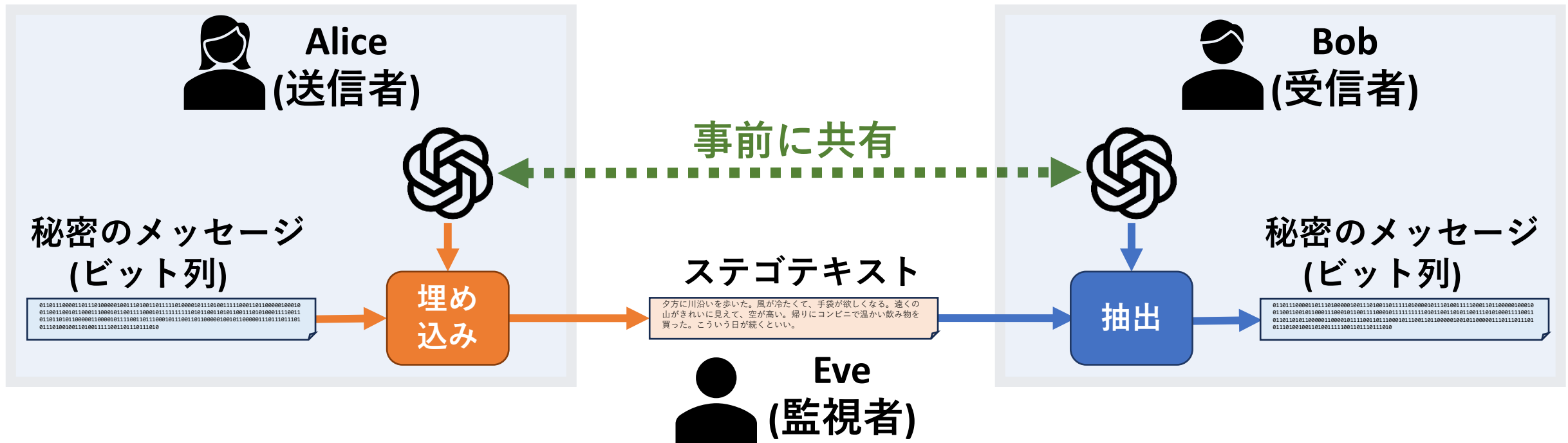
01101110000110111010000010011101001101111101000010111010011110001101100000100010  
0110011001011000111000101100111000101111111101011001100111010100011110011  
0110110110110000110000101110011011100010111001101100000100101100000111011101  
011101001001101001111100110111011010

# 言語ステガノグラフィ：タスク要件



1. **正確性:** Bobが秘密のメッセージを正確に復元できる

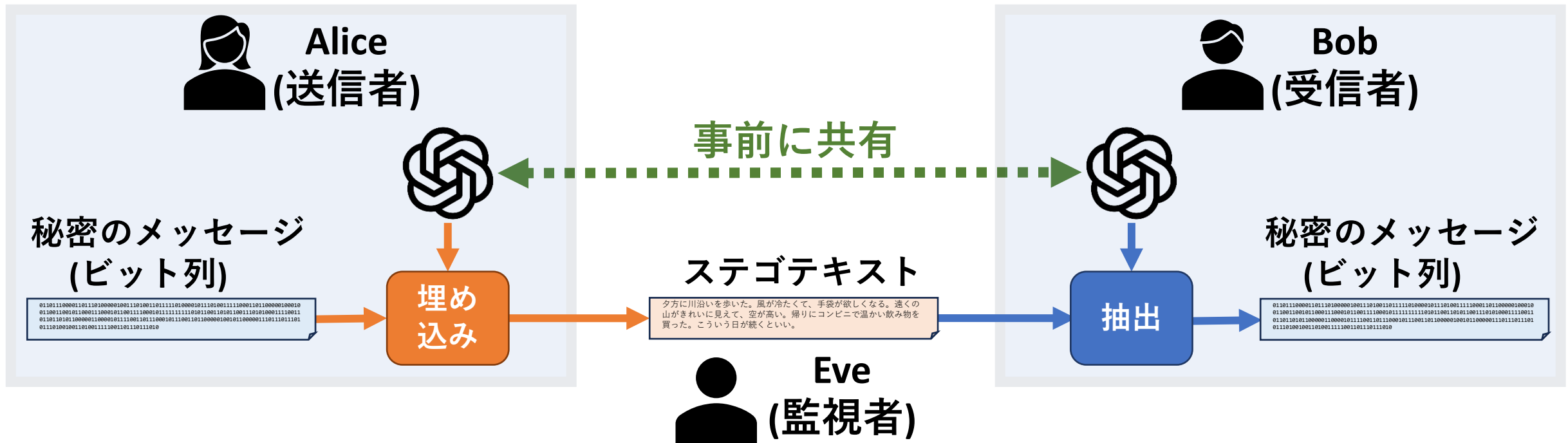
# 言語ステガノグラフィ：タスク要件



1. **正確性:** Bobが秘密のメッセージを正確に復元できる
2. **安全性:** Eveがステゴテキストを不審に思わない



# 言語ステガノグラフィ：タスク要件



1. **正確性:** Bobが秘密のメッセージを正確に復元できる
2. **安全性:** Eveがステゴテキストを不審に思わない
3. **埋め込み容量:** なるべく効率よく情報を送りたい

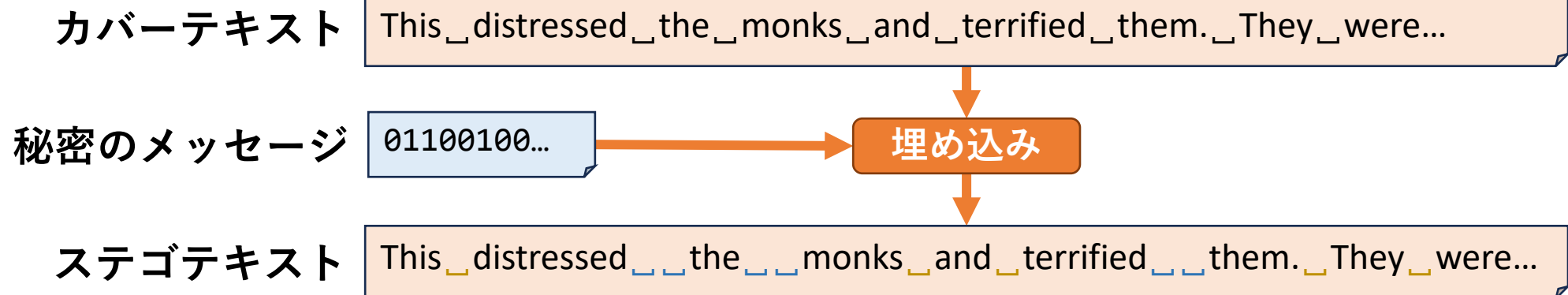
# 言語ステガノグラフィ：タスクまとめ

- 暗号との違い: 秘密のメッセージの**存在自体**を秘匿
- 関連技術
  - 電子透かし (watermarking): 生成物がAI由来か等を検出可能に
  - ステガナリシス (steganalysis): ステゴテキストを検出する技術
- 何のため？
  - 自由の技術的保証
  - 楽しいから☺

# (L)LM以前の言語ステガノグラフィ 1/2

空白を使った埋め込み [\[Bender+, 1996\]](#)

	0
	1



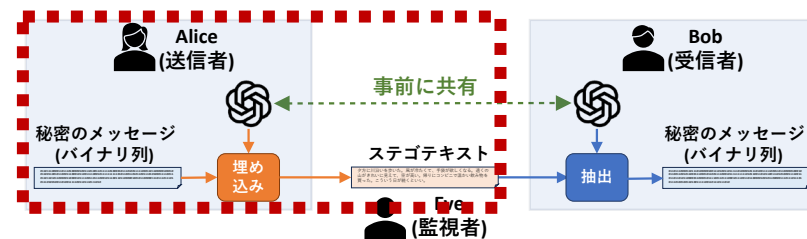
# (L)LM以前の言語ステガノグラフィ 2/2

同義語置換を使った埋め込み [\[Chang&Clark, 2014\]](#)

	1-bit Word			1-bit 2-bit Word		
	1-bit	Word		1-bit	2-bit	Word
We	0	<i>complete</i>	the charitable	0	00	<i>labor</i>
	1	<i>finish</i>		1	01	<i>project</i>
				0	10	<i>task</i>
				1	11	<i>undertaking</i>

- 多義性問題、文脈依存性問題 ...  
同義語セット {*bridge, span*} を *Suspension bridges are typically ranked by ...* に機械的に適用すると不自然に
- 手法の複雑化: 多義性を考慮した同義語ネットワーク、置換可能性のチェック ...
- [Ueoka+ \(2021\)](#) はマスク言語モデル (BERT) による候補列挙で手法を大幅に簡略化

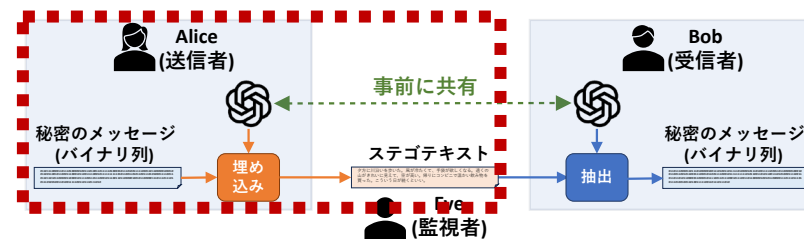
# LLMを用いた生成型言語ステガノグラフィ： [\[Fang+, 2017\]](#)以降 Aliceによる埋め込み



秘密のメッセージ: 100001 ...  
ステゴテキスト: :

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Aliceによる埋め込み

- 言語モデルに次の単語候補の確率を出力させる



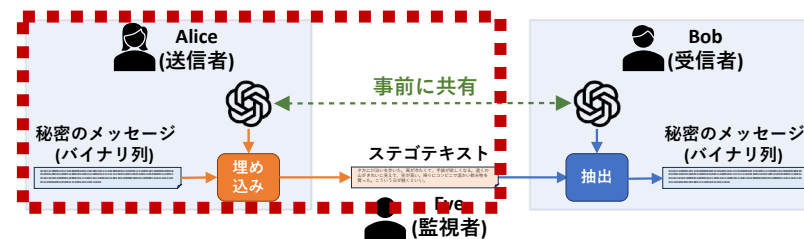
秘密のメッセージ: 100001 ...  
ステゴテキスト:

$P(w_1)$

単語	確率
今日	0.21
これ	0.20
私	0.15
また	0.13

# LLMを用いた生成型言語ステガノグラフィ：[Fang+, 2017]以降 Aliceによる埋め込み

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る

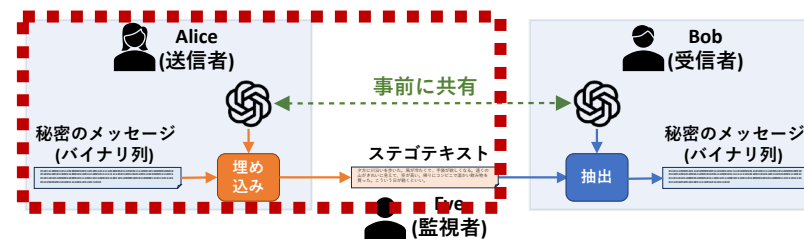


秘密のメッセージ: 100001 ...  
ステゴテキスト:

$P(w_1)$		
単語	確率	符号
今日	0.21	00
これ	0.20	01
私	0.15	10
また	0.13	11

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Aliceによる埋め込み

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- 秘密のメッセージのビット列に対応する単語を洗濯



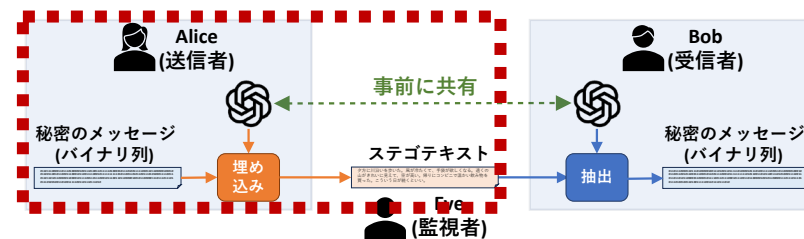
秘密のメッセージ: **10**0001 ...  
ステゴテキスト: 私

$P(w_1)$		
単語	確率	符号
今日	0.21	00
これ	0.20	01
私	0.15	10
また	0.13	11



# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Aliceによる埋め込み

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- 秘密のメッセージのビット列に対応する単語を洗濯

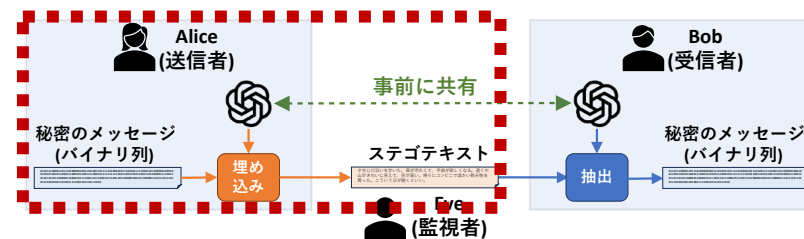


秘密のメッセージ: 100001 ...  
ステゴテキスト: 私

$P(w_1)$			$P(w_2 w_1)$		
単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00
これ	0.20	01	たち	0.21	01
私	0.15	10	の	0.20	10
また	0.13	11	から	0.13	11

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Aliceによる埋め込み

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- 秘密のメッセージのビット列に対応する単語を洗濯

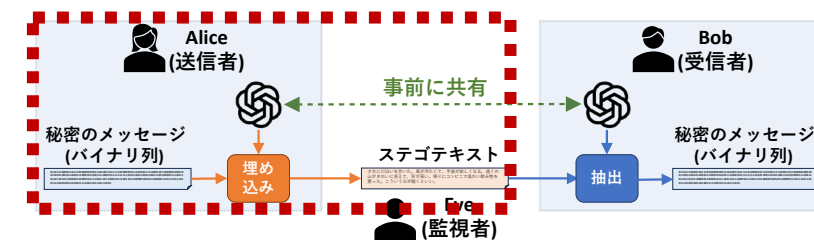


秘密のメッセージ: 10**00**01 ...  
ステゴテキスト: 私 は

$P(w_1)$			$P(w_2 w_1)$		
単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00
これ	0.20	01	たち	0.21	01
私	0.15	10	の	0.20	10
また	0.13	11	から	0.13	11

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Aliceによる埋め込み

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- 秘密のメッセージのビット列に対応する単語を洗濯

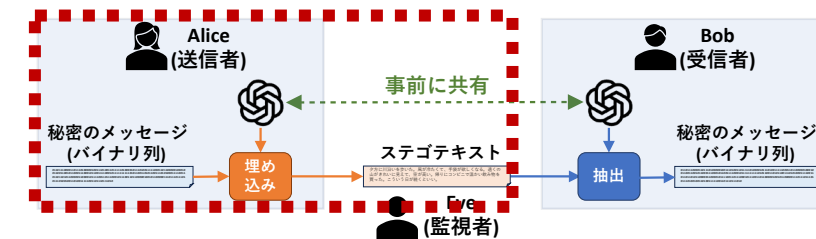


秘密のメッセージ: 100001 ...  
ステゴテキスト: 私 は

$P(w_1)$			$P(w_2 w_1)$		
単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00
これ	0.20	01	たち	0.21	01
私	0.15	10	の	0.20	10
また	0.13	11	から	0.13	11

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Aliceによる埋め込み

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- 秘密のメッセージのビット列に対応する単語を洗濯



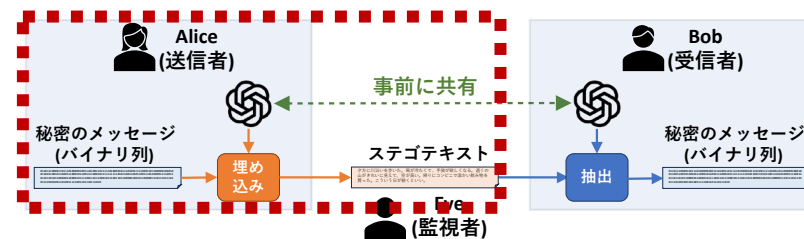
秘密のメッセージ: 100001 ...  
ステゴテキスト: 私 は

$P(w_1)$			$P(w_2 w_1)$			$P(w_3 w_1, w_2)$		
単語	確率	符号	単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00	今日	0.35	00
これ	0.20	01	たち	0.21	01	また	0.25	01
私	0.15	10	の	0.20	10	もっと	0.12	10
また	0.13	11	から	0.13	11	つい	0.09	11

...

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Aliceによる埋め込み

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- 秘密のメッセージのビット列に対応する単語を洗濯



秘密のメッセージ: 1000**01**...

ステゴテキスト: 私 は また ...

$P(w_1)$			$P(w_2 w_1)$			$P(w_3 w_1, w_2)$		
単語	確率	符号	単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00	今日	0.35	00
これ	0.20	01	たち	0.21	01	また	0.25	01
私	0.15	10	の	0.20	10	もっと	0.12	10
また	0.13	11	から	0.13	11	つい	0.09	11

...

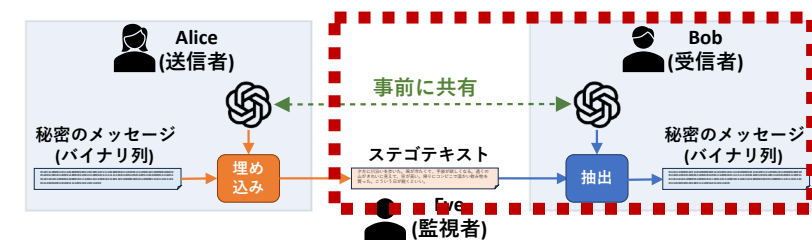
# LLMを用いた生成型言語ステガノグラフィ：[\[Fang+, 2017\]](#)以降 Bobによる抽出



ステゴテキスト: 私 は また ...  
秘密のメッセージ:

# LLMを用いた生成型言語ステガノグラフィ：[Fang+, 2017]以降 Bobによる抽出

- 言語モデルに次の単語候補の確率を出力させる



ステゴテキスト: 私 は また ...  
秘密のメッセージ:

$P(w_1)$	
単語	確率
今日	0.21
これ	0.20
私	0.15
また	0.13

# LLMを用いた生成型言語ステガノグラフィ：[Fang+, 2017]以降 Bobによる抽出

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る



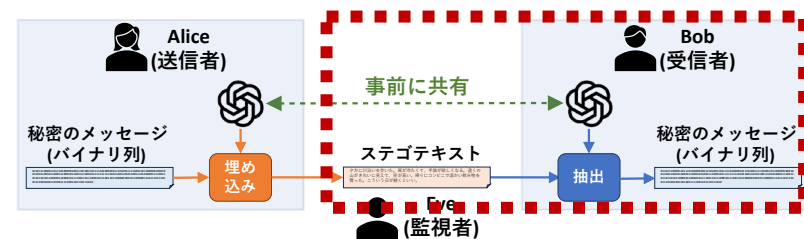
ステゴテキスト: 私 は また ...  
秘密のメッセージ:

$P(w_1)$		
単語	確率	符号
今日	0.21	00
これ	0.20	01
私	0.15	10
また	0.13	11



# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Bobによる抽出

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- ステゴテキストの次の単語に対応するビット列を選択

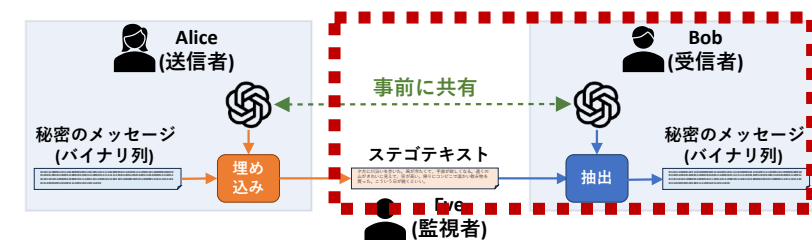


ステゴテキスト: 私 は また ...  
秘密のメッセージ: 10

$P(w_1)$		
単語	確率	符号
今日	0.21	00
これ	0.20	01
私	0.15	10
また	0.13	11

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Bobによる抽出

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- ステゴテキストの次の単語に対応するビット列を選択

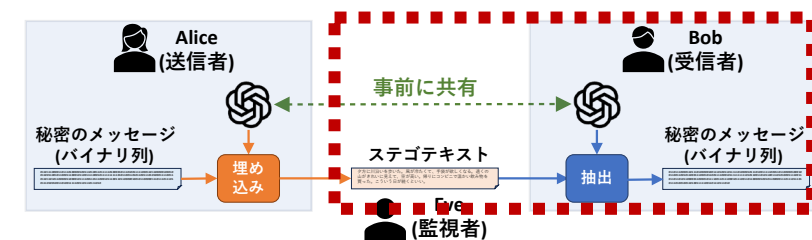


ステゴテキスト: 私 は また ...  
秘密のメッセージ: 10

$P(w_1)$			$P(w_2 w_1)$		
単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00
これ	0.20	01	たち	0.21	01
私	0.15	10	の	0.20	10
また	0.13	11	から	0.13	11

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Bobによる抽出

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- ステゴテキストの次の単語に対応するビット列を選択

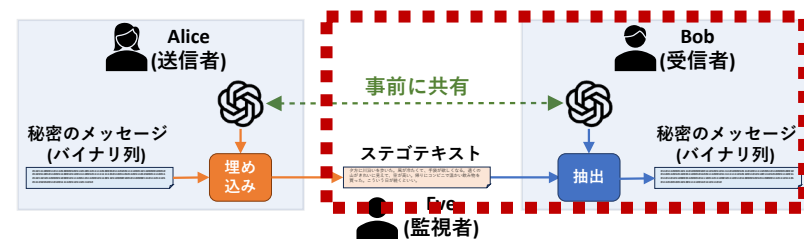


ステゴテキスト: 私 **は** また ...  
秘密のメッセージ: 1000

$P(w_1)$			$P(w_2 w_1)$		
単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00
これ	0.20	01	たち	0.21	01
私	0.15	10	の	0.20	10
また	0.13	11	から	0.13	11

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Bobによる抽出

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- ステゴテキストの次の単語に対応するビット列を選択



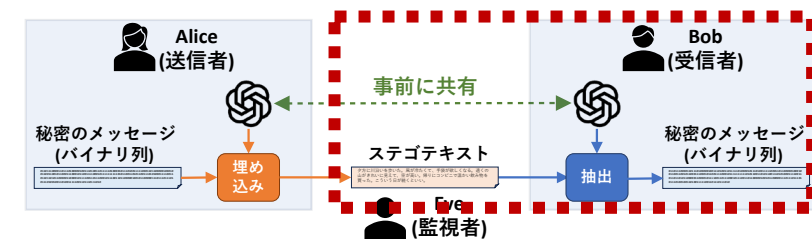
ステゴテキスト: 私 は また ...  
秘密のメッセージ: 1000

$P(w_1)$			$P(w_2 w_1)$			$P(w_3 w_1, w_2)$		
単語	確率	符号	単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00	今日	0.35	00
これ	0.20	01	たち	0.21	01	また	0.25	01
私	0.15	10	の	0.20	10	もっと	0.12	10
また	0.13	11	から	0.13	11	つい	0.09	11

...

# LLMを用いた生成型言語ステガノグラフィ: [\[Fang+, 2017\]](#)以降 Bobによる抽出

- 言語モデルに次の単語候補の確率を出力させる
- それらの単語にビット列を割り振る
- ステゴテキストの次の単語に対応するビット列を選択



ステゴテキスト: 私 は **また** ...  
秘密のメッセージ: 100001 ...

$P(w_1)$			$P(w_2 w_1)$			$P(w_3 w_1, w_2)$		
単語	確率	符号	単語	確率	符号	単語	確率	符号
今日	0.21	00	は	0.25	00	今日	0.35	00
これ	0.20	01	たち	0.21	01	また	0.25	01
私	0.15	10	の	0.20	10	もっと	0.12	10
また	0.13	11	から	0.13	11	つい	0.09	11

...

Year	Reference	Targeted metric(s)	Scenario(s)	Backbone model(s)	Training-free?	Asymmetric?	Black-box?
2017	Fang et al. [45]	Security	General	LSTM	✓	✓	✗
2017	Luo and Huang [103]	Efficiency	Lyric	LSTM	✗	✗	✗
2018	Yang et al. [233]	Security	General	Markov	✗	✗	✗
2019	Yang et al. [239]	Security	General	LSTM	✗	✗	✗
2019	Ziegler et al. [258]	Security	General	Model-agnostic	✓	✗	✗
2019	Dai and Cai [36]	Security	General	Model-agnostic	✓	✗	✗
2019	Yang and Ling [228]	Security	General	Model-agnostic	✓	✗	✗
2019	Tong et al. [164]	Efficiency	Lyric	LSTM	✗	✗	✗
2020	Yang et al. [230]	Security	General	LSTM	✗	✗	✗
2020	Yang et al. [235]	Security	General	GRU	✗	✗	✗
2020	Shen et al. [150]	Security	General	Model-agnostic	✓	✗	✗
2020	Kang et al. [71]	Security	General	LSTM	✗	✗	✗
2020	Wu et al. [199]	Security	General	Markov	✗	✗	✗
2020	Xiang et al. [209]	Efficiency	General	LSTM	✗	✗	✗
2021	Yang et al. [240]	Security	General	LSTM, BERT	✗	✗	✗
2021	Zhang et al. [250]	Security	General	Model-agnostic	✓	✗	✗
2021	Yang et al. [236]	Security	General	LSTM, GRU, Transformer	✗	✗	✗
2021	Kapchuk et al. [72]	Security	General	Model-agnostic	✓	✗	✗
2021	Zhang et al. [249]	Security	General	Transformer	✗	✗	✗
2021	Li et al. [88]	Security	General	LSTM, Transformer	✗	✗	✗
2021	Yang et al. [222]	Security	General	LSTM	✗	✗	✗
2021	Wang et al. [171]	Security	Lyric	LSTM	✗	✗	✗
2022	Yang et al. [238]	Security	Lyric	GRU	✗	✗	✗
2022	Yu et al. [244]	Security	General	Model-agnostic	✓	✗	✗
2022	Zhou et al. [257]	Security	General	LSTM, CNN	✗	✗	✗
2022	Yi et al. [241]	Security/Efficiency	General	BERT	✓	✗	✗
2022	Varol Arisoy [168]	Efficiency	General	LSTM	✗	✗	✗
2022	Adeeb and Kabudian [2]	Efficiency	Arabic text	LSTM	✗	✗	✗
2022	Nozaki and Murawaki [122]	Robustness	General	Model-agnostic	✓	✗	✗
2023	Sun et al. [158]	Security	General	T5	✗	✗	✗
2023	Lu et al. [101]	Security	General	Model-agnostic	✓	✗	✗
2023	Wang et al. [176]	Security	General	Transformer	✗	✗	✗
2023	Ding et al. [41]	Security	General	Model-agnostic	✓	✗	✗
2023	Wang et al. [177]	Security	General	BART, GPT-2	✓	✗	✗
2023	Wang et al. [178]	Security	Image-to-text	ClipCap	✗	✗	✗
2023	de Witt et al. [37]	Security	General	Model-agnostic	✓	✗	✗
2023	Yan et al. [221]	Security/Robustness	General	Model-agnostic	✓	✗	✗
2023	Lu et al. [102]	Security/Robustness	Translation	Transformer	✗	✗	✗
2023	Peng et al. [133]	Robustness	Image-to-text	ResNet-101, LSTM	✗	✗	✗
2024	Zhang et al. [247]	Security	Summarization	BERT, Conditional Random Field (CRF)	✓	✗	✗
2024	Wu et al. [198]	Security	General	Model-agnostic	✓	✓	✓
2024	Xiang et al. [203]	Security	Translation	BERT, LSTM	✗	✗	✗
2024	Lin et al. [94]	Security	General	Model-agnostic	✓	✗	✗
2024	Li et al. [89]	Security	General	Model-agnostic	✓	✗	✗
2024	Liu et al. [97]	Security	Live comments	ResNet, LSTM	✗	✗	✗
2024	Namitha et al. [118]	Security	General	Model-agnostic	✓	✗	✗
2024	Pang et al. [128]	Security	General	Model-agnostic	✓	✗	✗
2024	Li et al. [82]	Security	Group chat	BERT, Transformer	✓	✗	✗
2024	Cao et al. [25]	Security	General	Model-agnostic	✗	✗	✗
2024	Chen et al. [30]	Security	General	Model-agnostic	✗	✗	✗
2024	Qin et al. [141]	Security	General	Model-agnostic	✓	✗	✗
2024	Lin and Wang [95]	Security	Translation	LSTM	✗	✗	✗
2024	Yan et al. [220]	Security/Efficiency	General	ByGPT5	✓	✗	✗
2024	Steinebach [156]	Efficiency	General	Model-agnostic	✓	✓	✓
2024	Wang et al. [173]	Efficiency	Social media	Model-agnostic	✓	✗	✗
2024	Liao et al. [90]	Efficiency	Social media	Model-agnostic	✓	✗	✗
2024	Huang et al. [65]	Security/Robustness	General	Model-agnostic	✓	✗	✗
2024	Yan et al. [219]	Security/Robustness	General	Model-agnostic	✓	✗	✗
2024	Bai et al. [13]	Efficiency/Robustness	General	Model-agnostic	✓	✗	✓
2025	Wang et al. [181]	Security	General	Model-agnostic	✓	✗	✗
2025	Liao et al. [91]	Security	General	Model-agnostic	✓	✗	✗
2025	Woźniak et al. [194]	Security	General	Model-agnostic	✓	✗	✗
2025	Long et al. [99]	Security	General	Model-agnostic	✓	✗	✗
2025	Zhang et al. [251]	Security	General	Model-agnostic	✓	✗	✗
2025	Chen et al. [31]	Security	General	Model-agnostic	✓	✗	✗
2025	Shi et al. [151]	Security	General	XLM-RoBERTa, Transformer	✓	✗	✗
2025	Liu et al. [98]	Security	General	Model-agnostic	✓	✗	✗
2025	Kuznetsov et al. [78]	Security	General	Model-agnostic	✓	✗	✗
2025	Zhou et al. [256]	Security	General	Multi-agent	✓	✓	✗
2025	Pan et al. [125]	Security/Efficiency	General	Model-agnostic	✓	✗	✗
2025	Pang [126]	Security/Efficiency	General	Model-agnostic	✓	✗	✗
2025	Wang et al. [179]	Security/Efficiency	General	Model-agnostic	✓	✗	✗
2025	Bai et al. [12]	Security/Efficiency	General	Model-agnostic	✓	✗	✗
2025	Pang [127]	Efficiency	General	Model-agnostic	✓	✗	✓
2025	Huang et al. [66]	Efficiency	General	Model-agnostic	✓	✗	✗
2025	Wu et al. [200]	Efficiency	General	Diffusion language model	✗	✗	✗
2025	Li et al. [87]	Efficiency	General	Multi-agent	✗	✗	✓
2025	Farooq Ahmed Adeeb and Kabudian [46]	Efficiency	General	Model-agnostic	✓	✗	✗
2025	Yan and Murawaki [218]	Security/Robustness	General	Model-agnostic	✓	✗	✗
2025	Bai et al. [14]	Security/Robustness	General	Model-agnostic	✓	✓	✗
2025	Qi et al. [139]	Security/Robustness	General	Model-agnostic	✓	✗	✗
2025	Qi et al. [140]	Security/Robustness	General	Diffusion language model	✓	✗	✗
2025	Feng et al. [47]	Efficiency/Robustness	General	Model-agnostic	✓	✗	✗
2025	Perry et al. [136]	Robustness	General	Model-agnostic	✓	✗	✗
2025	Pang et al. [129]	Robustness	General	Model-agnostic	✓	✗	✗
2025	Guo et al. [57]	Robustness	General	Model-agnostic	✓	✗	✗
2025	Chen et al. [32]	Robustness	General	Model-agnostic	✓	✗	✗

# なぜ一向に実用化されない？ 1/2 [\[Yan+, 2025, preprint\]](#)

- 論文は量産されている
  - 主にセキュリティ分野で
- 「これを試して」と言える実装がない
  - ちゃんと動かない研究用コード @GitHub がほとんど

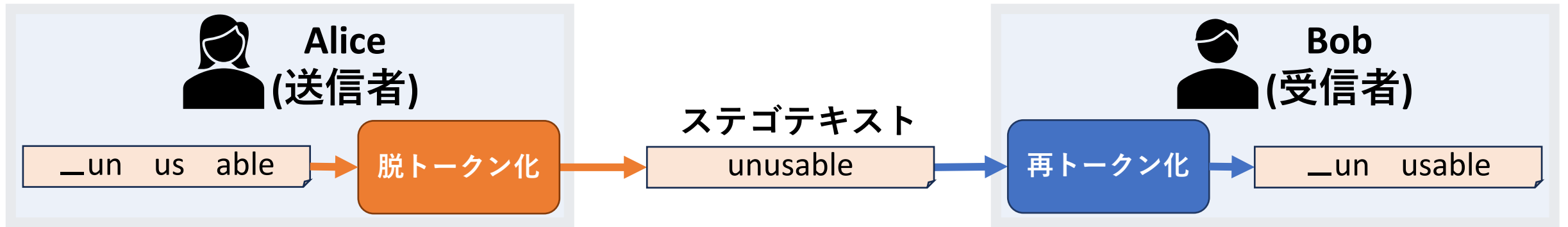
# なぜ一向に実用化されない？ 2/2 [\[Yan+, 2025, preprint\]](#)

依然として難しい点:

- **計算不確定性:** Aliceの確率計算をBobが厳密に再現できない
- **トークン化不整合性** (次のスライドで)
- **突然の終了:** 埋め込み完了時点でステゴテキスト生成を打ち切るのは明らかに不自然
- **LLMライセンス:** OpenRAILやLlamaは人をだます用途での利用をライセンスで禁止

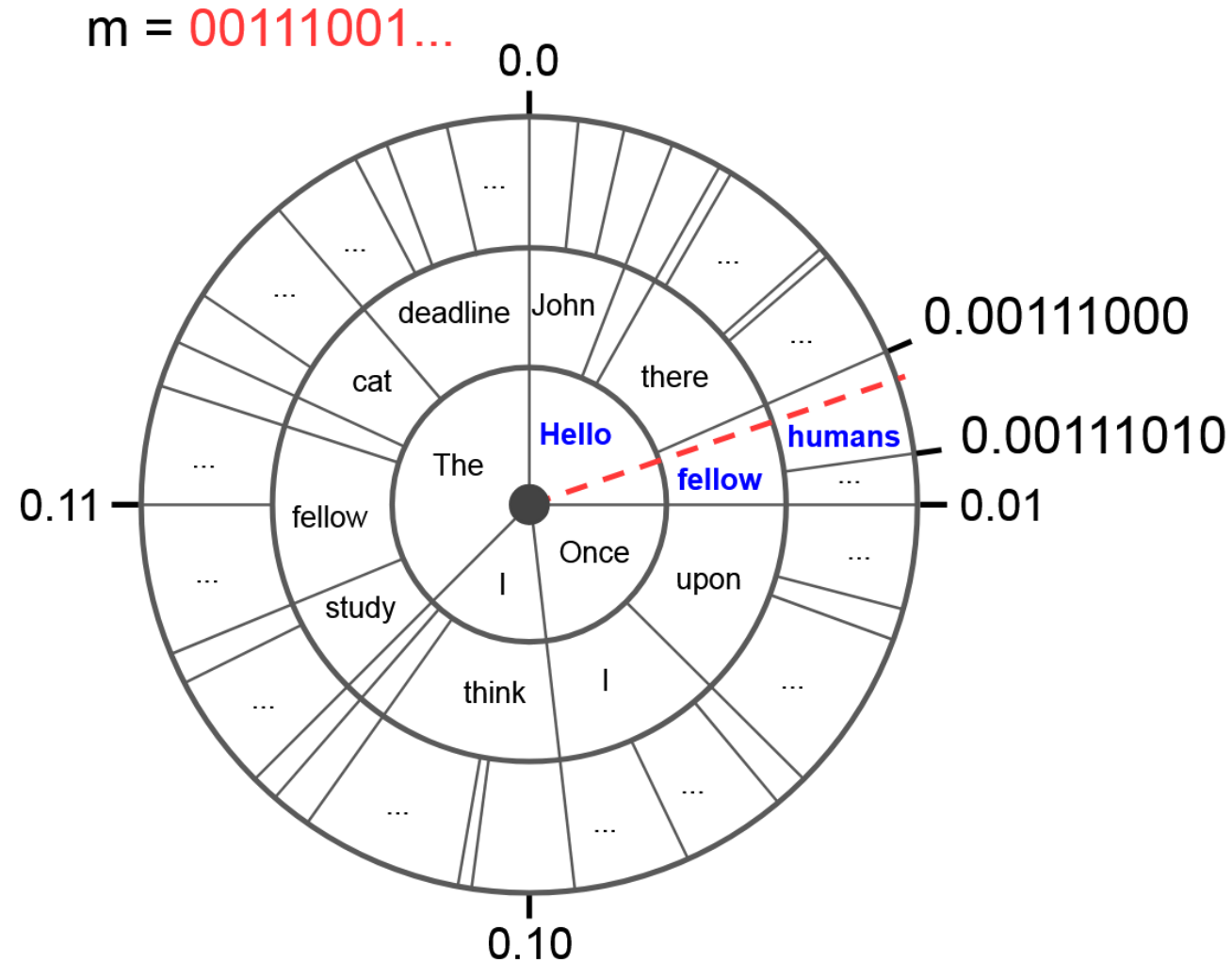


# トークン化不整合性



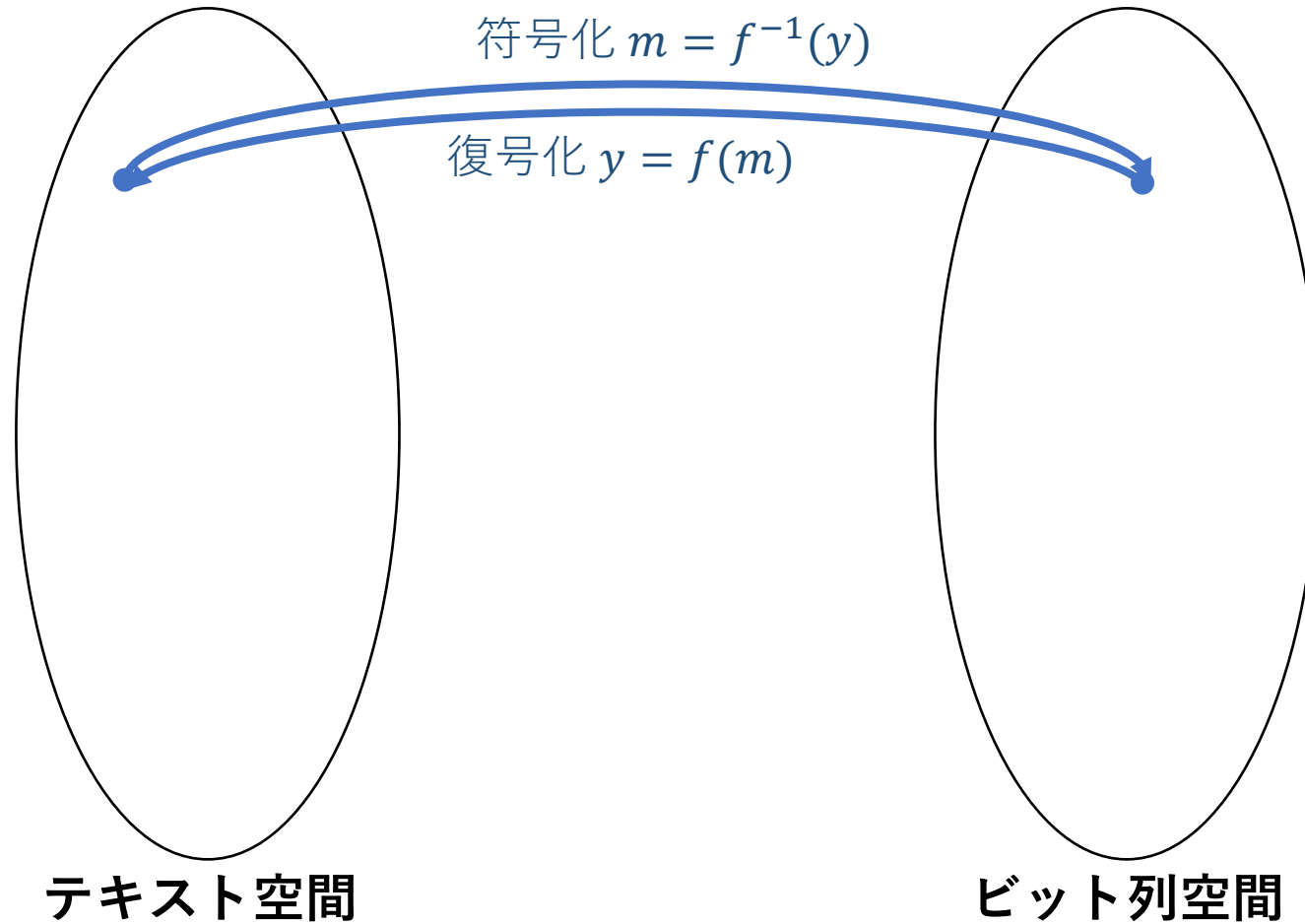
- 問題: **Eve**に怪しまれないように脱トークン化すると、**Bob**が元のトークン列を完全には再現できず、メッセージ抽出に失敗
  - サブワードの普及により、あらゆる言語が影響を受ける
- 解決策1: 候補列挙時の曖昧性解消+接頭辞分解による独自トークナイズ [\[Nozaki&Murawaki, 2022\]](#)
- 解決策2: **Alice**側で再トークン化確認+バックトラック [\[Yan&Murawaki, 2025\]](#)
- トークン化不整合性は解消されるが、安全性の理論的保証が失われる

# 算術符号 (Arithmetic Coding) [\[Ziegler+, 2019\]](#)

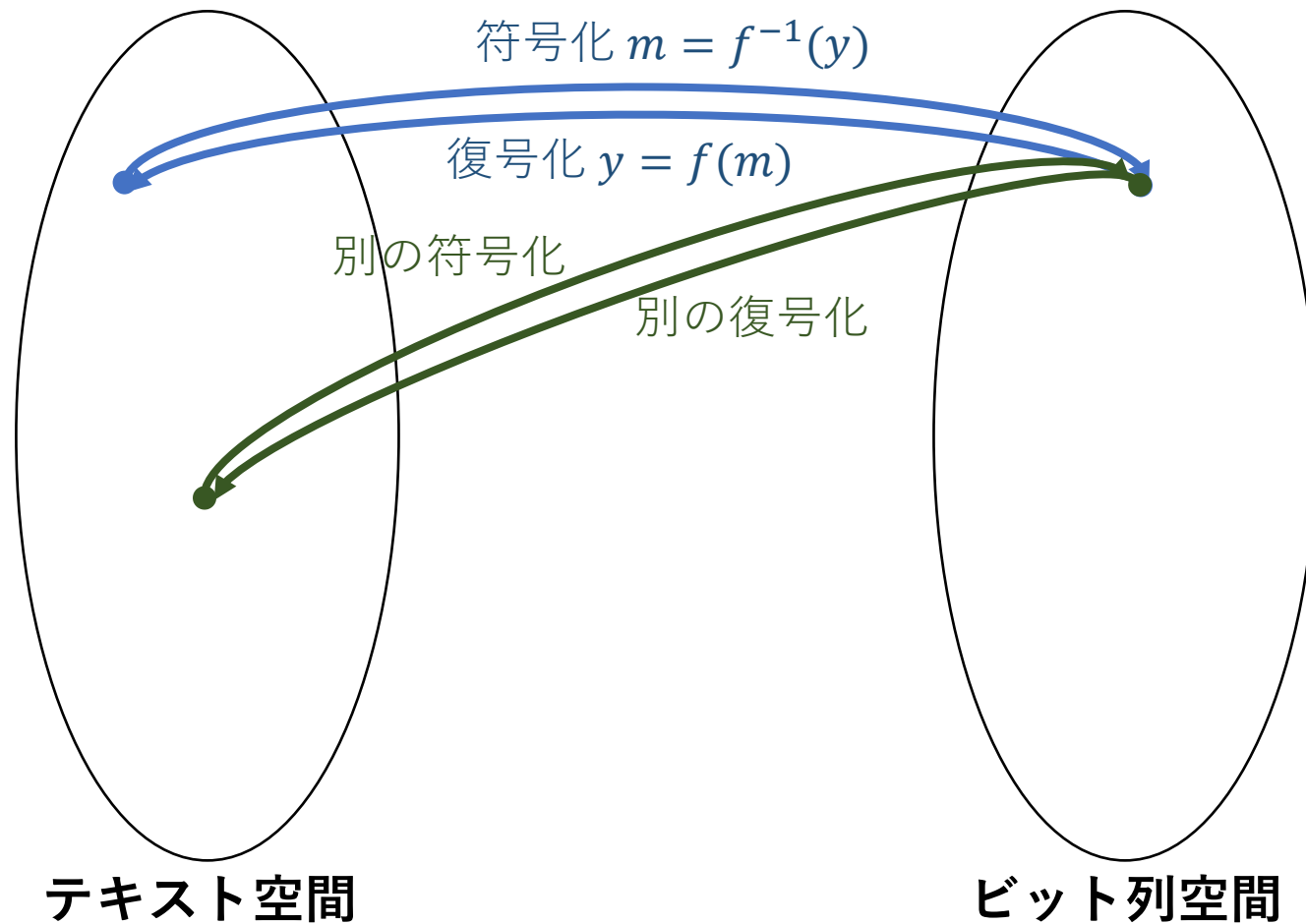


$y = f(m) = \text{"Hello fellow humans..."}$

# テキスト空間 $\leftrightarrow$ ビット列空間

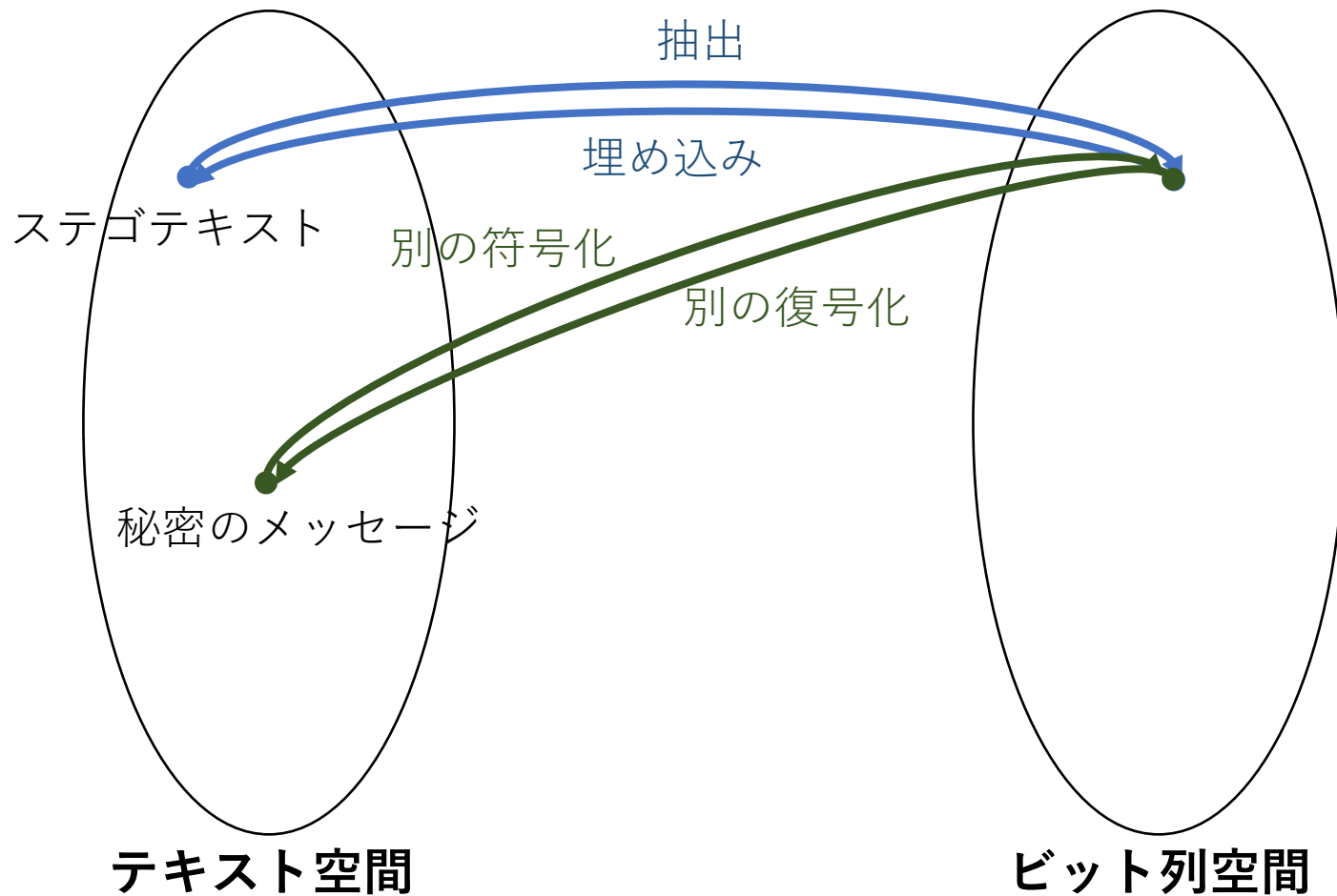


# テキスト空間 ↔ ビット列空間



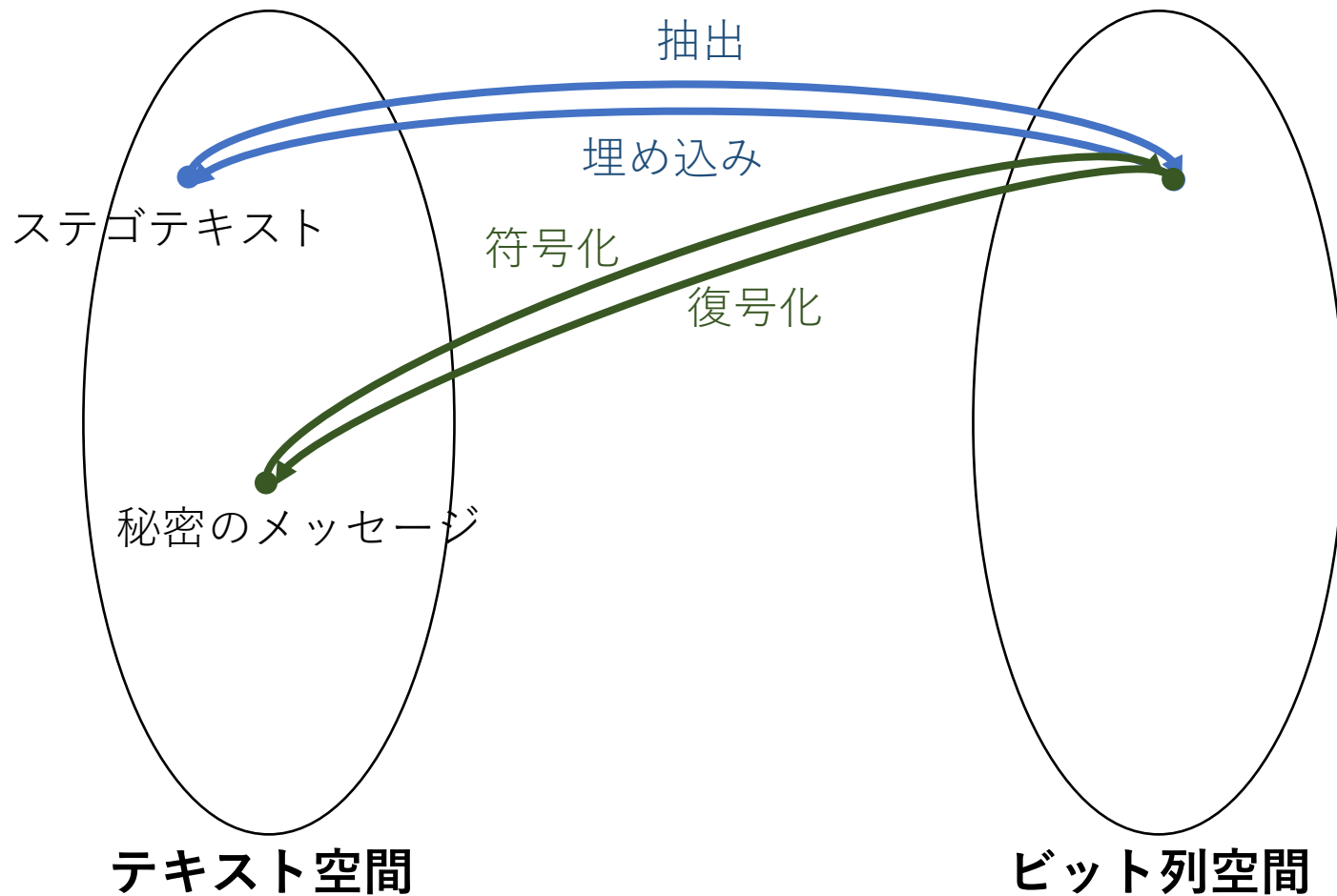
別の符号化/復号化:  
[textcoder@github](https://github.com/textcoder) は  
AES-GCM-SIVによる暗号化

# テキスト空間 ↔ ビット列空間



別の符号化/復号化:  
[textcoder@github](https://github.com/textcoder) は  
AES-GCM-SIVによる暗号化

# テキスト空間 ↔ ビット列空間



[Ziegler+ \(2019\)](#):

同じLLMでも与える文脈  
によって同じビット列に  
別のテキストが対応づく

**符号化/復号化:**

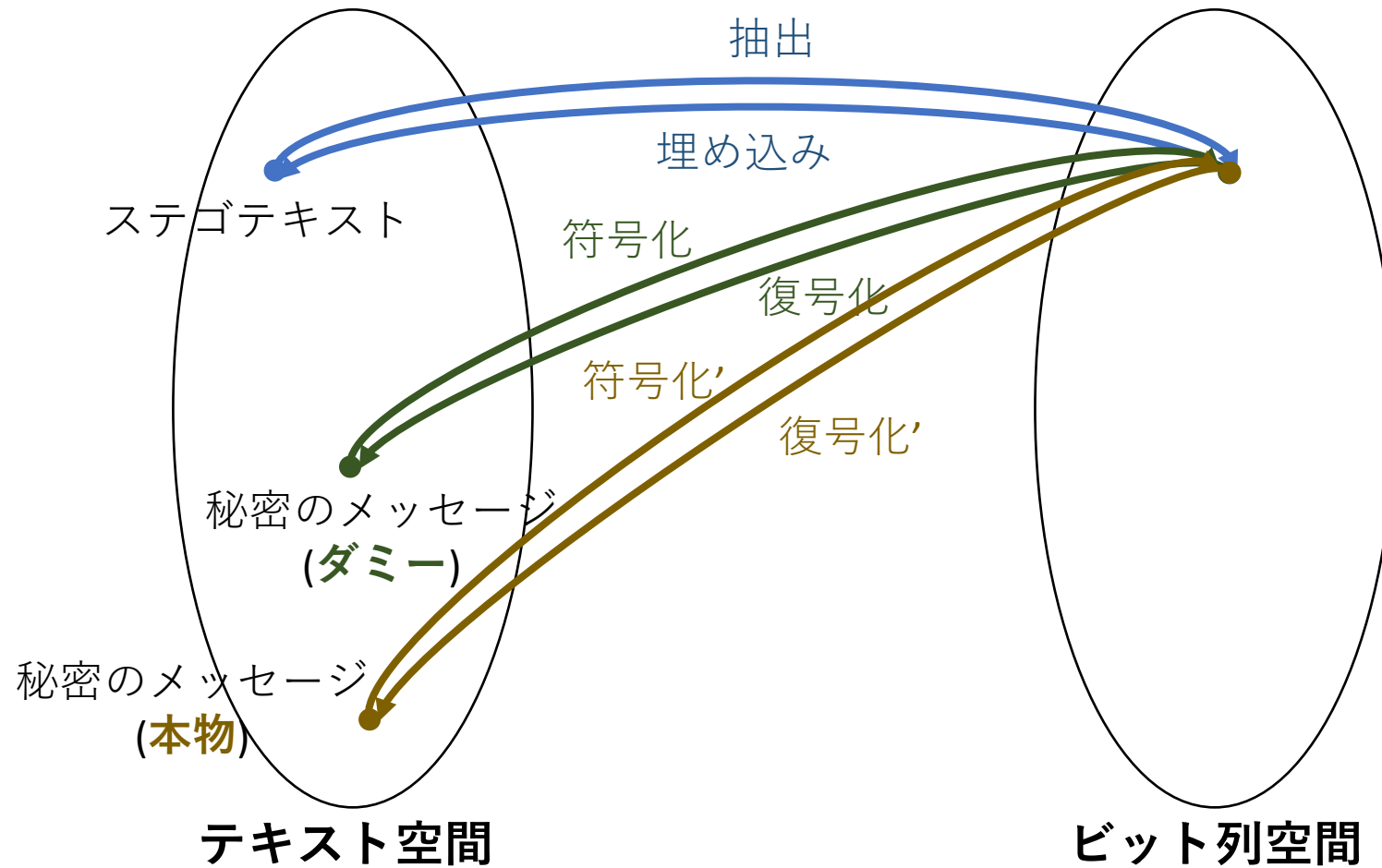
$$P_{\theta}(-|\phi)$$

**埋め込み/抽出:**

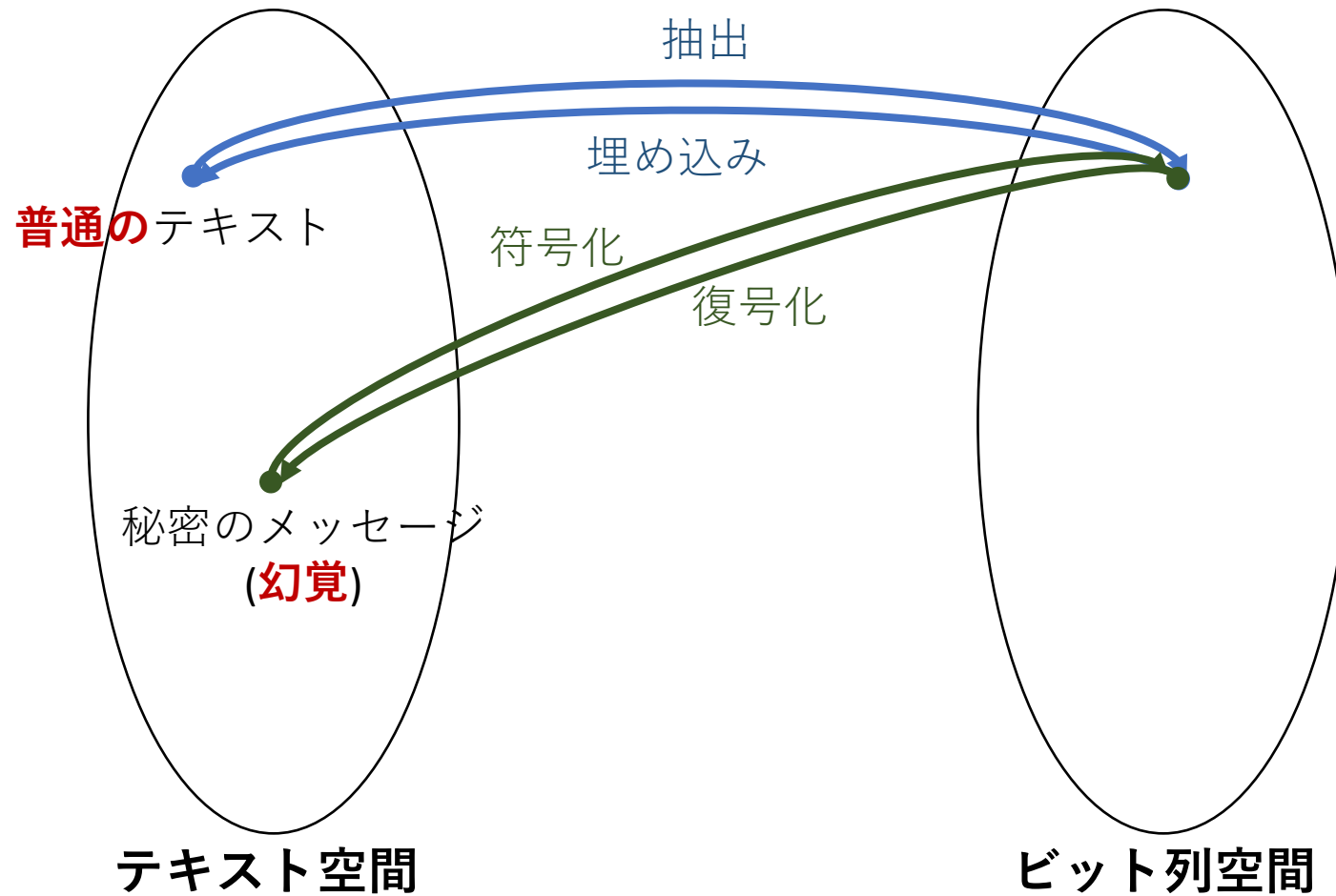
$$P_{\theta}(-|c)$$

$c$ はAliceとBobが事前に共有

# テキスト空間 ↔ ビット列空間



# テキスト空間 ↔ ビット列空間





# 言語ステガノグラフィ：まとめ

- テキストに秘密のメッセージを埋め込む技術
- LLMのおかげで主流は編集型から生成型に
- 細かい課題がいくつもあって依然として実用の壁は高い
- 「テキスト空間 $\leftrightarrow$ ビット列空間」という解釈と発展可能性

