Towards Near-imperceptible Steganographic Text

Falcon Z. Dai
Toyota Technological Institute at Chicago
Chicago, USA
dai@ttic.edu

Zheng Cai
Department of Computer Science
University of Colorado
Boulder, CO, USA
jon.z.cai@colorado.edu

Abstract

We show that the imperceptibility of several existing linguistic steganographic systems (Fang et al., 2017; Yang et al., 2018) relies on implicit assumptions on statistical behaviors of fluent text. We formally analyze them and empirically evaluate these assumptions. Furthermore, based on these observations, we propose an encoding algorithm called `patient-Huffman` with improved near-imperceptible guarantees.

1 Introduction

In recent years, we see many exciting developments in applied machine learning and, in particular, its application in the fundamental problem of language modeling (Sutskever et al., 2011; Jozefowicz et al., 2016) in the field of natural language processing (NLP). However, these advancements can be exploited by computationally resourceful entities such as a surveillance state to effectively monitor its citizens' *ostensibly* private communications at scale.

We are motivated to study the communication privacy problem of concealing sensitive messages in monitored channels. In order to avoid raising suspicion in the monitoring party, we want to hide the intended message inside a fluent message, known as a *stegotext*, indistinguishable from what is expected in such channels. This is a problem studied primarily in *steganography* and steganography researchers have a keen interest in linguistic steganography as it presents fundamental challenges (Chang and Clark, 2014); the linguistic channel carries few bits per symbol on average (Shannon, 1951; Brown et al., 1992) making it hard to hide a message. In contrast, images and sound recordings have a high information theoretic entropy comparing to a written message making it relatively easy to embed a message in the noise floor of the channel.

This problem of hiding secret messages in plain sight might evoke spy stories of concealing messages on newspaper advertisements during Cold War. Such manual methods have been superseded by algorithmic approaches. Classic methods prior to the advance of applied machine learning in this domain typically try to produce *grammatical* English with generative grammar (Chapman and Davida, 1997). However, such generation methods fall short in terms of *statistical* imperceptibility (Meng et al., 2008). This makes them vulnerable to automated detection. Generating *fluent*[1] text at scale is at the heart of the steganography problem, and language models (LM) studied in NLP provide a natural solution by letting us draw samples of fluent texts.

At the working heart of a LM-based stegosystem, there lies an encoding algorithm that encodes a ciphertext (a random string indistinguishable from a series of fair coin flips) into a fluent stegotext using an LM. From the communication standpoint, this encoding must be uniquely decodable, i.e. different ciphertext are encoded into different stegotexts otherwise the receiver will not be able to decode and recover the ciphertext. Instead of sampling according to the LM, an encoding algorithm effectively induces a new language model by providing a non-standard way to draw samples from the LM. Thus, from the language modeling standpoint, in order to achieve statistical imperceptibility, extra care is needed to ensure the resulting LM is close to the original LM (Sec. 2.2). Various uniquely decodable algorithms has been devised by recent pioneering works (Fang et al., 2017; Yang et al., 2018) leveraging recurrent neural network-based LMs, and the high-quality stegotexts generated show tremendous promise in terms of both flu-

[1] It is often referred to as "naturalness" in linguistic steganography literature.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4303–4308
Florence, Italy, July 28 - August 2, 2019. ©2019 Association for Computational Linguistics

ency and information hiding capacity. However, these methods do not explicitly provide guarantees on imperceptibility. Instead, their imperceptibility, as we will argue, relies on *implicit* assumptions on the statistical behaviors of the underlying LM, and ultimately, of fluent texts (Sec. 3). We will empirically evaluate these assumptions and show that they are problematic (Sec. 3.1). In response, we will propose an improved encoding algorithm `patient-Huffman` that explicitly maintains imperceptibility (Sec. 3.3).

To see that imperceptibility crucially depends on the statistics of fluent texts, consider plausible continuations of the following two prefixes, "I like your" and "It is on top." In the first case, there are many likely next words such as "work", "style", "idea", "game", "book", whereas in the latter, there are few such as "of", ",", "and", "." with "of" being overwhelmingly likely. Intuitively speaking, the distribution over next tokens in fluent texts is sometimes uniform and sometimes highly concentrated.[2] When it is concentrated, if we choose the next token by flipping fair coins, we will be sampling from a very different distribution and risk being detected after a few samples. In `patient-Huffman`, we actively evaluate how different the encoding distribution and the LM distribution are, and avoid encoding at steps that can expose us.

The highlights of this work are the following:

- We quantify statistical imperceptibility with total variation distance (TVD) between language models. We study the TVD of several encoding algorithms and point out the implicit assumption for them to be near-imperceptible.
- We use a state-of-the-art transformer-based, subword-level LM, `GPT-2-117M` (Radford et al., 2019), to empirically evaluate the plausibility of assumptions implicitly made by different encoding methods.
- We propose an encoding algorithm `patient-Huffman` with strong relative statistical imperceptibility.

2 Formalism

Suppose Alice (sender) wants to send Bob (receiver) a sensitive message (plaintext) through a channel monitored by Eve (adversary). This channel may be shared by many other communicating parties. Furthermore, Eve expects to see fluent natural language texts in this channel. Alice wants to avoid sending non-fluent texts through this channel to raise Eve's suspicion while ensuring that only Bob can read the sensitive message.

One class of solutions is to

1. Alice *encrypts* the plaintext message into a ciphertext with a key shared with Bob.[3]
2. Alice *hides* the ciphertext, which has the statistics of random coin flips, into a fluent stegotext.
3. Alice sends the stegotext through a channel monitored by Eve.
4. Bob receives the stegotext and *seeks* the ciphertext from it.
5. Bob *decrypts* the ciphertext with the shared key and obtain the plaintext message.

Linguistic stegosystems concern with steps 2 (hide) and 4 (seek), i.e. encoding a random bitstring into a fluent stegotext and extracting the original bitstring from such fluent stegotexts, respectively.

A *vocabulary* Σ of size V is a finite set of tokens.[4] An *extended vocabulary* Σ^* is the set of all finite sequences of tokens from Σ. We call its elements *texts*. A *language model* ℓ is a measure over some extended vocabulary Σ^*. Furthermore, we assume that we have access to the conditional distribution over the next token given a prefix $\mathbb{P}[s_{t+1}|s_1 \cdots s_t; \ell]$ and the distribution of the initial token $\mathbb{P}[s_1; \ell]$. An LM specified in this way allows us to draw samples easily. We can draw a sample text by drawing each s_t one at a time for $t = 1, 2, \cdots$ according to LM. We call the random sample text $s := s_1 \cdots s_T \sim \ell$ an *ℓ-fluent text*.

Total variation distance (TVD) between two measures p and q over the same events denoted by σ-algebra $\mathcal{F}$, is $d(p, q) := \sup_{E \in \mathcal{F}} |p(E) - q(E)|$ (see A.1 for more facts).

A *ciphertext* b of length C is a random variable $b := b_1 b_2 \cdots b_C \sim \text{Bernoulli}(1/2)^C$. An *encoding algorithm* $\mathfrak{A}_\ell$ is an injective map from ciphertexts

[2] Under the estimates of `GPT-2-117M`, the first continuation has entropy of 11.2 bits and the latter, 0.43 bits. The most likely next tokens shown are also drawn from this model.

[3] Public key encryption can also work. Alice will encrypt the plaintext with Bob's public key and Bob decrypts with his private key in that case.

[4] Tokens can be characters, subword units or words depending on the modeling choices. We will be using subword units based on byte pair encoding in our experiments.

to distributions over texts $\mathfrak{A}_\ell : \{0,1\}^C \to \Delta(\Sigma^*)$ which may depend on the LM ℓ. Injectivity ensures that the stegotexts are unique decodable.

2.1 Near-imperceptibility

Instead of using the informal notion of imperceptibility common in steganography literature which relies on a human eavesdropper (playing Eve) judging the quality, we consider a formal statistical notion of near-imperceptibility. We say a measure over texts p, i.e. an LM, is *δ-imperceptible with respect to a language model ℓ* if $d(p,\ell) < \delta$. This formalization is motivated by the fact that for any algorithm, it takes *at least* $\Omega\left(\frac{1}{\delta^2}\right)$-many samples to tell whether the samples come from ℓ or p with high confidence.[5] The smaller $d(p,\ell)$ is, the more samples are required for Eve to discover the presence of our steganographic communication regardless of her computational resource. Therefore, we want to find encoding algorithms that are near-imperceptible with respect to the true LM of the monitored channel.

2.2 Decomposition of TVD

Suppose the true LM of the monitored channel is ℓ^*, and we have access to a base LM ℓ, then running encoding algorithm $\mathfrak{A}_\ell$ induces an effective LM $\mathfrak{A}[\ell] := \mathbb{E}_b[\mathfrak{A}_\ell(b)]$. Consider the TVD between the effective LM and the true LM

$$d(\ell^*, \mathfrak{A}[\ell]) \le d(\ell^*, \ell) + d(\ell, \mathfrak{A}[\ell])$$

by triangle inequality.

The first term on the right hand side corresponds to how good our LM ℓ is, which is limited by the advancement in LM research. The second term is the gap due to our encoding algorithm and it is the focus of this study. Without knowing how large the first term is, we can still pursue a meaningful *relative* imperceptibility guarantee of the form, “it will be as hard to detect the presence of the steganographic communication as detecting the presence of ℓ-fluent texts.”

We can further decompose the second term on the right hand side over each generation step and suppose $s_{<t} := s_1 \cdots s_{t-1}$ is the prefix, we can use Pinsker's inequality (Tulsiani, 2014) and additivity of KullbackLeibler divergence (KL divergence)[6] over product measures to obtain a bound via the KL divergence on each step

$$d(\ell, \mathfrak{A}[\ell]) \le \sqrt{\frac{\ln 2}{2} \sum_{t=1}^{\infty} D_{KL}\left(\mathbb{P}[\cdot|s_{<t};\ell] \,||\, \mathbb{P}[\,\cdot\,|s_{<t};\mathfrak{A}[\ell]]\right)}.$$

Hence in order to obtain relative near-imperceptibility, it is sufficient to ensure that at each generation step, the effective LM $\mathbb{P}[\,\cdot\,|s_{<t};\mathfrak{A}[\ell]]$ is close to the base LM $\mathbb{P}[\cdot|s_{<t};\ell]$. (See an analogous decomposition in terms of per-step TVD in A.1.)

3 Analysis

Suppose $h \in \Sigma^*$ is a prefix (tokens generated up to the current step), and the base LM is ℓ.

3.1 `Bins`

Fang et al. (2017) divide the vocabulary into 2^k disjoint bins of equal sizes, $\{B_1, \cdots, B_{2^k}\}$, that is, $\Sigma = \sqcup_{i=1}^{2^k} B_i$ and $|B_i| = V/2^k$. The partition is randomly chosen and shared between Alice and Bob. Then we split a ciphertext into (C/k)-many length-k blocks $a_1 \cdots a_{C/k}$. We encode the ciphertext by encoding each a_i. To encode a random block $a \sim \text{Bernoulli}(1/2)^k$, we pick a token from the a-th bin, i.e. B_a, according to ℓ. Suppose s falls in the bin B^s, we effectively sample a token s according to

$$\mathbb{P}[s|h;\texttt{Bins}[\ell]] = \frac{1}{2^k} \frac{\mathbb{P}[s|h;\ell]}{\mathbb{P}[B^s|h;\ell]}$$

and the KL divergence is

$$D_{KL}(\mathbb{P}[\cdot|h;\ell] || \mathbb{P}[\cdot|h;\texttt{Bins}[\ell]]) = k - H(B).$$

(See A.3 for detailed derivation.) The last term is the entropy of the partitions at the current step which is bounded between zero and k. Hence, the KL divergence is at most k at each step. However, if the probability mass is roughly evenly distributed over each of the 2^k bins, then the KL divergence is close to zero. This is the *implicit* assumption about fluent texts `Bins` makes.

We empirically examine how well this assumption holds. We use `GPT-2-117M` as the base LM and sample from it 50 prefixes with 40 steps each, saving 2K steps of conditional distributions. We fix a randomly chosen partition of $2^3 = 8$ bins. The computed KL divergence concentrates in the low-bit region with a second mode near 3 bit, the

[5] This is a basic result from information theory. See for example (Tulsiani, 2014).

[6] We will consistently compute KL divergence in base 2, i.e. we measure entropy in bits.

maximum (Fig. 1). The mean of the distribution is 0.7 bits, meaning that in ten steps the KL bound on TVD will be vacuous, encoding about 30 bits of ciphertext.

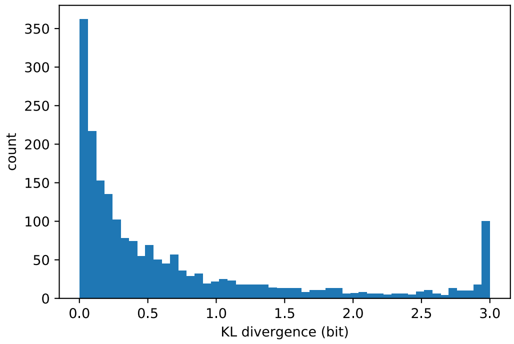

Figure 1: $D_{KL}(\mathbb{P}[\cdot|h;\ell]||\mathbb{P}[\cdot|h;\texttt{Bins}[\ell]])$ in bits over a sample of 2K tokens generated from `GPT-2-117M` with $2^3 = 8$ bins. Fewer tokens with high bits is better.

3.2 Variable-length coding (VLC)

Instead of using a fixed-length coding (one stegotext token always encodes k bits in `Bins`), `VLC` encodes one or more bits per generated token (Yang et al., 2018). `VLC` constructs a Huffman coding of Σ at each step according to $\mathbb{P}[\cdot|h;\ell]$.[7] Then we sample a token from the constructed Huffman tree c by following the bits in ciphertext starting at the root, taking the left subtree if the bit b_i is zero else the right subtree until reaching a leaf. The resulting Huffman distribution m_c assigns probability mass $1/2^r$ for a token at depth r. Being a minimum redundancy code, the corresponding Huffman distribution has the minimum KL divergence among binary prefix-free codes (Huffman, 1952) of at most 1 bit. But will there be steps with large KL divergence like the example "It is on top" in Sec. 1? We computed the KL divergence of Huffman codes for the same 2K samples (Fig. 2). The mean of 0.12 bits is significantly lower than `Bins`'s but it still has a second mode near 1 bit, the maximum.

3.3 patient-Huffman

We improve `VLC` further by explicitly checking if the TVD[8] (or the KL divergence) between the base LM distribution and the Huffman distribution is small enough (Algorithm 1). If the TVD is greater than a specified threshold at the current encoding step, instead of sampling from the Huffman distribution, we sample from the base LM distribution and patiently wait for another opportunity.

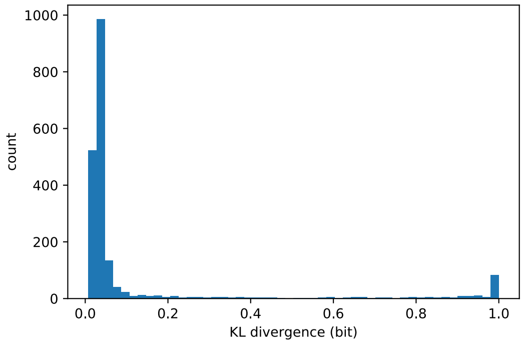

Figure 2: $D_{KL}(\mathbb{P}[\cdot|h;\ell]||\mathbb{P}[\cdot|h;\texttt{VLC}[\ell]])$ in bits over a sample of 2K tokens generated from `GPT-2-117M`. Fewer tokens with high bits is better.

Clearly, this ensures that each step incurs no more additional TVD than the specified threshold δ. In principle, if we set $\delta_t = o(1/t)$ for the t-th step, then we can bound the total TVD, *guaranteeing* the relative near-imperceptibility of the generated stegotext.

However, in practice, getting any meaningful bounds (total TVD $\ll 1$) will require setting very small δ_t and this translates to an empirical assumption that *many* fluent texts' next token distributions lie *arbitrarily* close to the Huffman distributions. Examining Fig. 2, we see that there are many steps

Algorithm 1 patient-Huffman (one encoding step)

```
Input: a language model $\ell$, prefix $h \in \Sigma^*$, an imperceptibility threshold $\delta$, a ciphertext $b$.
Return: a stegotext from $\Sigma^*$.
Compute the distribution of the next token $p \leftarrow \mathbb{P}[\cdot|h;\ell]$.
Construct a Huffman tree $c$ for $p$.
Compute the TVD (or the KL divergence) between $p$ and the corresponding Huffman distribution $m_c$.
if TVD (or KL divergence) $< \delta$ then
    Decode a token $w$ by consuming the ciphertext $b$ and following its bits starting at the root of Huffman tree $c$.
else
    Sample a token $w$ according to $p$.
end if
Append the token to prefix $h \leftarrow h; w$
return $h$
```

[7] This takes $O(V \log V)$.

[8] Computing TVD or KL divergence is $O(V)$.

with KL divergence close to zero. This assumption, though more benign than VLC's or Bins's empirically, is hard to establish theoretically for fluent text.

4 Discussion

We focus on the encoding algorithm in our analysis but it is not hard to see that Bob can correctly decode the ciphertext from the stegotext by running the same algorithm with the same LM and the same ciphertext block size (and other parameters if any) as Alice, e.g. patient-Huffman with the same threshold, and extract the unique (Huffman) code corresponding to the observed token as ciphertext.

The generic approach of embedding a ciphertext into a stegotext that has some anticipated distribution studied in this paper can very well apply to other channels such as images or audios where we can access the marginal distribution via a (deep) generative model.

Formal notions of steganographic secrecy have been studied in the cryptography community. In particular, Hopper et al. (2008) develop a complexity theoretic notion and characterize its necessary conditions and its maximum bandwidth under a perfect sampling oracle. This is stronger than our setting where a trained LM provides us an approximate access to the marginal distribution. The information theoretic notion of imperceptibility we proposed independently is most similar to the notion of steganographic security in (Cachin, 2004). Further study connecting these results is needed. Of particular interest is an extension called robust steganography, where an *active* adversary may alter messages, e.g. by injecting typographical errors. The stegosystems studied here are vulnerable to such attacks.

OpenAI's decision of making GPT-2-117M publicly available enables our empirical studies and it likely will for other studies. However, this released trained version is inferior to the full GPT-2 model (Radford et al., 2019). While we appreciate OpenAI's general precaution and specific arguments against its release, we want to note, with this work, that its release can also offer social good by enhancing communication privacy. We advocate for the public release of strong trained models as a way to mitigate the disparity in access to both data and computational resources.

Lastly, the full implementation of the stegosystem proposed in this work is made open-source under a permissive license.[9]

Acknowledgments

We thank the anonymous reviewers for their suggestions. We thank David MacAllester for a helpful discussion on Huffman coding. We thank Avrim Blum for bringing related works in the cryptography community to our belated attention.

References

Peter F Brown, Vincent J Della Pietra, Robert L Mercer, Stephen A Della Pietra, and Jennifer C Lai. 1992. An estimate of an upper bound for the entropy of english. *Computational Linguistics*, 18(1):31–40.

Christian Cachin. 2004. An information-theoretic model for steganography. *Information and Computation*, 192(1):41–56.

Ching-Yun Chang and Stephen Clark. 2014. Practical linguistic steganography using contextual synonym substitution and a novel vertex coding method. *Computational Linguistics*, 40(2):403–448.

Mark Chapman and George Davida. 1997. Hiding the hidden: A software system for concealing ciphertext as innocuous text. In *International Conference on Information and Communications Security*, pages 335–345. Springer.

Tina Fang, Martin Jaggi, and Katerina Argyraki. 2017. Generating steganographic text with lstms. In *Proceedings of ACL 2017, Student Research Workshop*, pages 100–106.

Nicholas Hopper, Luis von Ahn, and John Langford. 2008. Provably secure steganography. *IEEE Transactions on Computers*, 58(5):662–676.

David A Huffman. 1952. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

Peng Meng, Liusheng Huang, Zhili Chen, Wei Yang, and Dong Li. 2008. Linguistic steganography detection based on perplexity. In *2008 International Conference on MultiMedia and Information Technology*, pages 217–220. IEEE.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. (Accessed on 2019-4-23).

[9] https://github.com/falcondai/lm-steganography. We also include generated samples and illustrative examples.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1715–1725.

Claude E Shannon. 1951. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024.

Madhur Tulsiani. 2014. Pinskers inequality and its applications to lower bounds. Lecture Notes.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Zhongliang Yang, Xiaoqing Guo, Ziming Chen, Yongfeng Huang, and Yu-Jin Zhang. 2018. Rnnstega: Linguistic steganography based on recurrent neural networks. *IEEE Transactions on Information Forensics and Security*.