Linguistic Steganography Using a Transformer-based Causal Language Model

Arian Tashakkor, *40023494*
Shahab Mohrehkesh, *40032134*

Abstract

This is the report for the final project of the Advanced Information Retrieval course. In this endeavor, we implement a linguistic steganography scheme using GPT-2 as a causal language model. We will then compare our results with two previous works previously mentioned as a part of the project's proposal, namely: GAN-TStega and an Attention-LSTM based method. We demonstrate that our method produces better than or comparable results than the aforementioned works.

I. Introduction

A. Language modeling

LANGUAGE modeling (LM) is the use of various statistical and probabilistic techniques to determine the probability of a given sequence of words occurring in a sentence. Language models analyze bodies of text data to provide a basis for their word predictions [1].

Language models determine word probability by analyzing text data. They interpret this data by feeding it through an algorithm that establishes rules for context in natural language. Then, the model applies these rules in language tasks to accurately predict or produce new sentences. The model essentially learns the features and characteristics of basic language and uses those features to understand new phrases.

A good language model should also be able to process long-term dependencies, handling words that may derive their meaning from other words that occur in far-away, disparate parts of the text. An LM should be able to understand when a word is referencing another word from a long distance, as opposed to always relying on proximal words within a certain fixed history.

B. Transformers

Attention allows a language model to make predictions by looking at the entire input (not the most recent segment) and selectively attend to some parts of it. This selection is determined by a set of weights that are learned during training. For example, "The fox saw a rabbit. It was very hungry so it tried to grab it but it dodged just in time". The attention mechanism can be used to figure out what word each "it" in the input sequence refers to [2]. A Transformer is a type of neural network architecture. Transformers are models with an encoder-decoder structure that make use of the attention mechanism. The encoder component encodes the input data by selectively attending to different parts of the input using the attention mechanism and passes the encodings to the decoder to be decoded. Figure 1 shows the overview of a transformer.

Like recurrent neural networks (RNNs), transformers are designed to process sequential input data, such as natural language, with applications towards tasks such as translation and text summarization. However, Unlike recurrent neural networks, e.g LSTMs, transformers read all the input words at once, eliminating the need to wait for the first 3 words to be passed through until the fourth word can be read for instance. This allows for more parallelization than RNNs and therefore reduces training times [2].

This led to the development of pretrained systems such as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer), which were trained with large language datasets, such as the Wikipedia Corpus and Common Crawl, and can be fine-tuned for specific tasks.

C. Transformer-based Language model

GPT-2 is a large transformer-based language model with 1.5 billion parameters, trained on a dataset of 8 million web pages. GPT-2 is trained with a simple objective: predict the next word, given all of the previous words within some text. More precisely, inputs are sequences of continuous text of a certain length and the targets are the same sequence, shifted one token (word or piece of word) to the right. The model internally uses a mask-mechanism to make sure the predictions for the token i only uses the inputs from 1 to i but not the future tokens [4].

This way, the model learns an inner representation of the English language that can then be used to extract features useful for downstream tasks. The diversity of the dataset causes this simple goal to contain naturally occurring demonstrations of many tasks across diverse domains.In addition to its incredible language generation capabilities, it is also capable of performing tasks like question answering, reading comprehension, summarization, and translation.

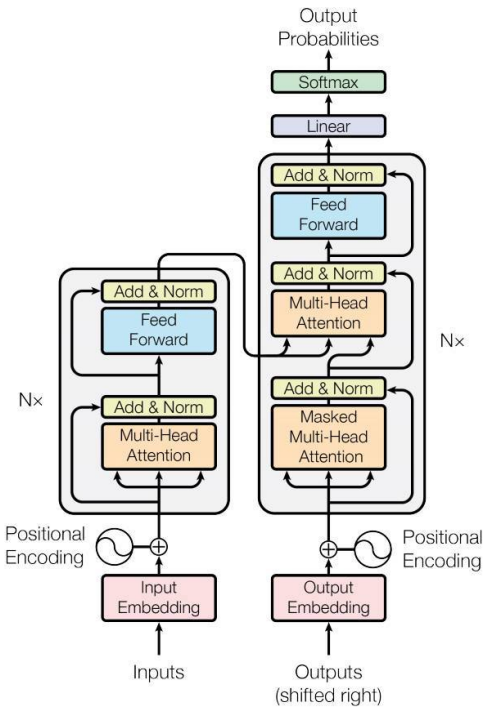


Fig. 1: The architecture of a transformer. [3]

D. Steganography

Steganography is the art and science of embedding secret messages in a cover message in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message [5]. Figure 2 depicts a basic steganographic model.

Steganography can be used to conceal almost any type of digital content, including text, image, video or audio content; the data to be hidden can be hidden inside almost any other type of digital content.

E. Text Steganography

Text steganography, is a mechanism of hiding secret text message inside another text as a covering message or generating a cover message related with the original secret message. There are three main categories used to hide text-in-text messages, that is, format based, random and statistical generations, and linguistic method [6], [7].

- **Format based**
  In this method, the covering message will not be changed in case of its words and sentences; modifications will just be made on the spaces between the words, lines, or/and paragraphs using special characters, that is, white-space steganography. In using this method, the number of added special character (i.e., white space) to the other spaces in the cover message depends on the value of the binary string digits. This method is not without its flaws. If the stego file is opened with a

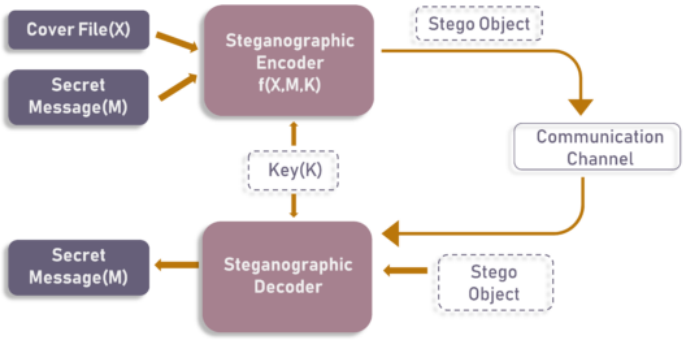


Fig. 2: Steganographic model. [5]

	GAN TStega	Attention LSTM
$k = 1$	a cat sitting on top of a wooden bathroom tub. a motorcycle parked in front of a temple. a pine apple in a corner.	I am a freshman, now I want to go to graduate school, want to be a lawyer, want to ask your opinion, is how to arrange their own schedule and efficiency? I am a more serious student, but always remember in class. I want to make good use of myself when I can't remember the foreign language. The efficiency is very low. So I also want to ask, what should I do? thank you.
$k = 2$	a safety conscious rock outside of formation in the night direction. a blurry mirror cuts a sink and medicine pen. a narrow bathroom with a mirror mirror looking toilet behind some ruins.	I want to lose weight! I am a senior three student, because I want to lose weight and get in shape, but now I am on a diet and weigh 110. What can you eat if you want to lose weight now? Otherwise I don't feel like eating anything. I won't eat it. Don't know what to do to get results?

TABLE I: Comparison of generated texts by Attention LSTM and GAN TStega for different embedding rates.

word processor, misspellings and additional white spaces will get identified. Changed fonts sizes can excite suspicion to a human reader. Moreover, if the initial plaintext is accessible, comparing this plaintext with the suspected steganographic text can create manipulated element of the text quite visible.

- **Random and statistical generation methods**
  These methods automatically generate the cover text message; it does not need an existing cover message. The generated cover message uses the secret message in generation process. This algorithm uses this language structure and properties—i.e., how to create the sentences, what is the past format of a verb, etc. Also, these methods use grammar to produce suitable cover message. In this type of text steganography, an extra complexity is added (time and space) to generate a full paragraph; this consumes long time to embed and extract the secret message in/from the cover message [8].
- **Linguistic methods**
  This method is used to hide a message in another message depending on the linguistic structure of the cover message (the punctuation marks) or the words semantic. In order to hide a message, the cover message must have punctuation marks (i.e., comma, full stop, etc.) to hide the secret message in behind. These punctuation marks are the identification sign for the secret message. This type of steganography uses the synonyms of each word to hide the message; this method is done by searching for synonyms for each word in the secret message to generate the output cover message [8].

II. Related Work

In this section we will discuss two other linguistic steganography approaches that have been previously explored in the literature. Seeing as how linguistic steganography, and text steganography as a whole, is a fairly new field, the major breakthroughs in it didn't happen until very recently. We will briefly discuss two leading papers in the field of linguistic steganography.

First, in [9], a linguistic steganography approach is discussed using a Generative Adversarial Network (GAN), where a GAN is employed to create a causal language model - i.e., a language model that is able to predict the next token, given a sequence of previous context tokens up to now. The details of their approach are mostly impertinent to our effort save for the encoding-decoding scheme that we will later discuss at length when explaining our proposed method.

Moreover, in [10] a similar method using an Attention LSTM language model for the Chinese language is offered that achieves significantly better results than the previous work. In this paper, a similar encoding-decoding scheme is used with the added novelty of using keywords to guide the process of text generation and create higher quality text. In table I, we can see a side by side comparison of generated texts from both methods at different embedding rates.

III. Proposed Method

We will now discuss our proposed method in further detail. Figure 3 represents a high-level view of the architecture of our proposed method. We begin by discussing the coding scheme, followed by a brief discussion about our implementation and datasets, and finally we introduce the metrics that we use for the evaluation of our work.

A. Steganographic Coding Scheme

In order to encode and decode the bits embedded within the text, we use a similar coding scheme to the one described in [9] with several minor modifications. The main idea which allows for encoding bits within units of text is that a causal language model trained on a fairly extensive vocabulary is able to infer the semantic relationships between words and is thereby able to produce a ranking of next possible tokens given a context where the first few words in the ranking are either synonymous, or appear in such a way that the continuation of the context is not jeopardized with any of them. For example, consider the context "My dog is very". A good causal LM will produce a ranking for the next token given this context in such a way that

the high ranking tokens are all valid for the continuation of the sentence, for instance, the ranking [‘happy’, ‘joyous’, ‘sad’, ‘sick’, ...]. As we can see, the two highest ranking tokens are ‘happy’ and ‘joyous’ which are synonymous, yet the rest of the sentence can be meaningful with tokens at positions three and four, namely ‘sad’ and ‘sick’, as well.

If this assumption holds true for some causal LM, we can then use the index of the next token that has been chosen to continue the generated text as a means of hiding secret data bits inside the text.

Encoding:

Given a “begin phrase” and “k”, the encoding scheme can be described as follows:

1) Read the secret message file in bit-wise fashion.
2) Split the bits into k-sized chunks.
3) Convert each binary chunk into its decimal representation (n).
4) Starting from the begin phrase as context, repeat until reaching the end of message:
   a) Calculate a ranking of the top 2^k most probable next tokens given previous context.
   b) If the highest ranked token is a whitespace or a punctuation, add it to the context and iterate.
   c) Else, choose the n-th most probable next token and add it to the context.
5) If the context doesn’t end with an *endofsentence*, keep generating and adding to the context until an *endofsentence* token.

Step 4-b from the encoding algorithm ensures that we produce high quality stego text by not encoding any bits inside punctuations and whitespaces this helps keep the integrity and the structure of the text intact from a grammatical standpoint. Moreover, step 5, ensures that the stego text always terminates meaningfully and with an *endofsentence* token (here just a simple ‘.’).

Decoding:

The decoding scheme is the exact dual of the encoding scheme. It takes as input a coded “stego text” and the “begin phrase” and “k” (here it is assumed that the message length in bits is known to the decoder):

1) Initialize reconstructed message.
2) Initialize a pointer to the first word of the stego text (cur_ptr).
3) Starting from the word pointed to by cur_ptr as context, repeat until length of reconstructed message is equal to length of secret message:
   a) Calculate a ranking of the top 2^k most probable next tokens given previous context.
   b) If the token at $cur\_ptr + 1$ is a whitespace or a token, set $cur\_ptr = cur\_ptr + 1$ and iterate.
   c) Else, find the index of the token pointed to by cur_ptr in the ranking (n).
   d) Convert n to its binary representation and zero-pad it to k bits if necessary.
   e) Append the binary representation to the reconstructed message. Set $cur\_ptr = cur\_ptr + 1$.

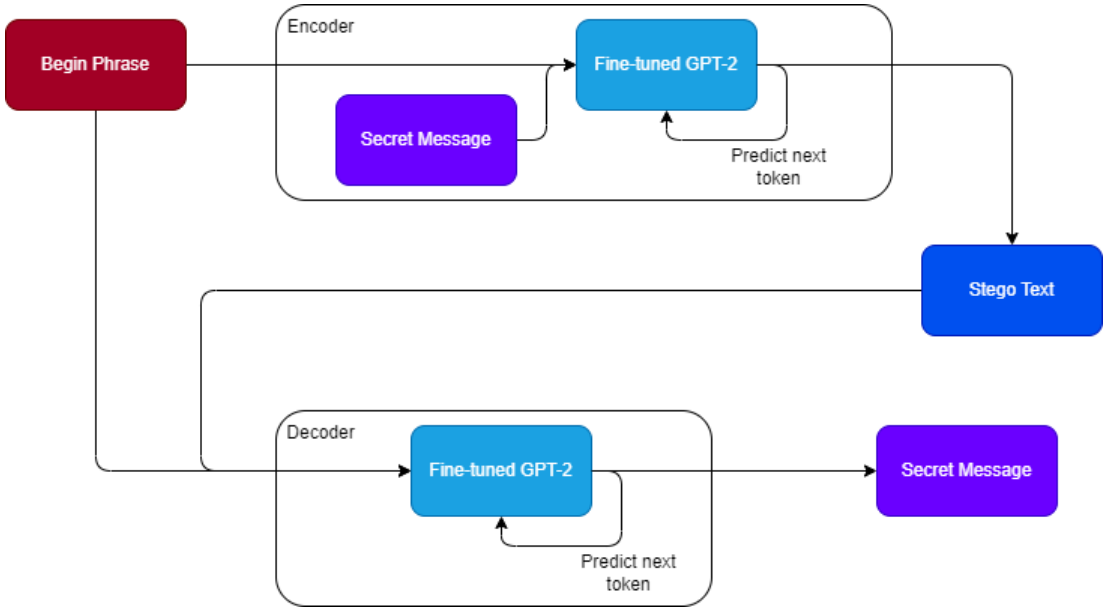
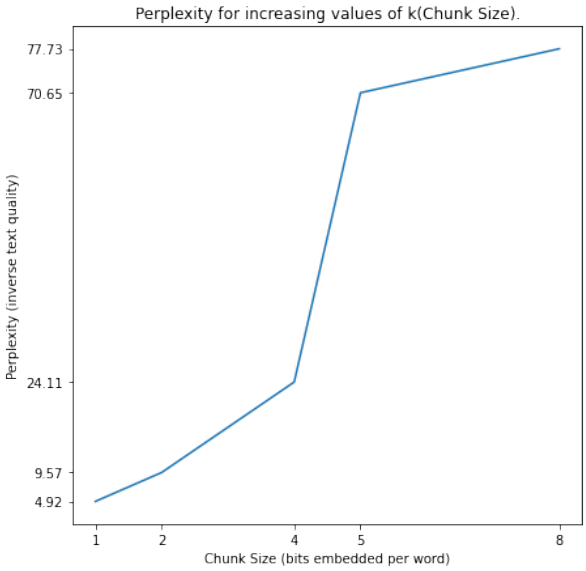
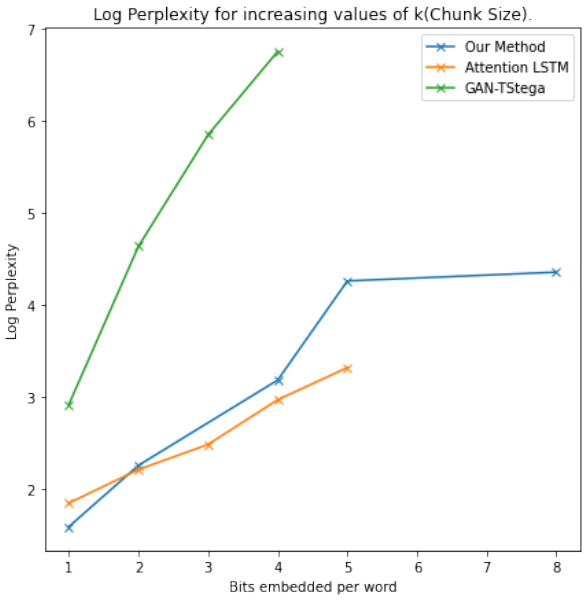


Fig. 3: Our framework architecture.



(a) Average perplexity of our method.



(b) Log Perplexity in comparison with other methods.

Fig. 4: Experiments

B. Implementation

Our implementation uses the 'PyTorch' deep learning framework and the Hugging Face 'transformers' library [11] to fine-tune the medium version of GPT-2 [12] on the same datasets used in [9]. [1] We use Image COCO captions and EMNLP WMT17 source dataset as the training dataset. Image COCO is the dataset for image captioning. COCO Captions contains over one and a half million captions describing over 330,000 images. EMNLP WMT17 is a dataset for machine translation from which we selected the English news data to fine-tune our model [13], [14].

GPT-2 Medium was fine-tuned on this dataset on a free-of-charge instance of a K80 NVIDIA GPU on Google Colab for one epoch at a learning rate of $1e-6$ which took just over 4 hours to finish.

C. Evaluation

In order to ensure the concealment of steganographic text and evaluate the quality of the produced text the statistical difference between steganographic distribution and real distribution should be as small as possible [9]. Here, we use Perplexity to measure the difference between two statistical distributions which is defined as follows:

$$\text{Perplexity}\left(T_{gen}\right) = 2^{-\frac{1}{N}\sum_{i=1}^{m}\log_2 p(w_i)} \tag{1}$$

where w_i, is the i-th symbol of the generated text. Perplexity in essence denotes the crossentropy of the two distributions.

To compare the proposed method with the previous methods, and due to a lack of unsupervised generated text quality evaluation methods, the semantic comparison of the generated texts as well as Perplexity is used.

IV. Experiments and Results

In order to test the viability of our method, we created a dummy 200-bit secret message file which we then encoded with k=[1,2,4,5,8] with 7 different begin phrases. We then calculated the average perplexity for each k value over different begin phrases. The results can be seen in figures 4a and 4b.

We can observe that the perplexity of the generated text increases as the number of bits embedded within it grow larger. This is expected as embedding k bits within each unit of text means that we will have to produce a ranking of the next 2^k tokens to accommodate all the possible values encoded by k bits. The larger k is, the bigger 2^k and the higher the chances of having to choose a lower ranking next token which causes the quality of the text to deteriorate in exponential proportion to the choice of k.

Moreover, we can see our method outperforms GAN-TStega by a very significant margin and is able to produce comparable results to the Attention-LSTM method. Please refer to the Appendix to see a list of our partial generated texts which you can then compare with table I.

[1]This report is accompanied by an implementation of our method which you can find on the file *IR_project_final.ipynb*.

APPENDIX

SAMPLES OF GENERATED TEXT AT DIFFERENT VALUES OF k AND DIFFERENT BEGIN PHRASES

Tables II and III show partial generated samples at k values 2, 4 and 8 for begin phrases "Moreover" and "There" respectively. All of the texts encode the same secret message. Only the beginning of the samples are shown here for brevity.

As we can see, the samples generated are fairly high quality. Grammatical guidelines seem to have been followed and correct punctuations are put whenever necessary. Additionally, we observe that as the value of k increases, the samples deteriorate in quality according to our expectation and yet the length of stego text also decreases because more bits are embedded within each word.

Also noteworthy is that the length of stego text is variable because we let the algorithm skip over punctuations and whitespaces and additionally keep generating until it reaches an *endofsentence* token.

Begin Phrase = Moreover	
$k = 2$	Moreover, it's important to remember, this is just the tip. For example, in the above example, the user has the ability of setting the background color, but the background is not visible. In the above example, if a user is viewing an app with the same background image, but different background image, then it will not work. [rest is omitted]
$k = 4$	Moreover, we also find an example (Section 2.3) of using some simple arithmetic operators to represent a number. The above code uses arithmetic and string comparisons. If two arguments were provided, the compiler must choose among one and zero. [rest is omitted]
$k = 8$	Moreover, how appropriate its passage does portray L.C. MacEaw. —see Epoll, Par. VII. Note.[113] Perhaps: either s. 5, it is a "fellow-worker" of the Lord, or s.

TABLE II: Samples generated starting with "Moreover"

Begin Phrase = There	
$k = 2$	There's no way to tell what kind of impact the bill would make in the state. But if it does, we can't wait any longer. This story has been updated. Copyright © 2017, The Daily Caller News Service. We welcome feedback. If your area would benefit from some civil discussion, please email tips@wcj.org. We also encourage you read our Comment Policies. [rest is omitted]
$k = 4$	There will not come back," a visibly frustrated Obama said. But when I spoke later with his son, who was with his mom, they insisted the fight would only go deeper. "This fight is really only getting harder as more women get pregnant and get to a time in life when they're not able to have children," said Obama's daughter, Malia.
$k = 8$	There would then continue, through multiple versions of time, as 'universe' came down to be seen with more of everything. 'At one very pivotal junction, this changed. If, over an ere short lifetime, space-borne particles were to be observed, they would be seen as a single, unified entity, with no room for variation.

TABLE III: Samples generated starting with "There"

ACKNOWLEDGMENT

The authors would like to thank Dr. Alireza Basiri for their invaluable guidance over the course of the semester.

REFERENCES

[1] "What is Language Modeling?" [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/language-modeling
[2] M. Fallah, "An Overview of Different Transformer-based Language Models," Mar. 2021. [Online]. Available: https://techblog.ezra.com/an-overview-of-different-transformer-based-language-models-c9d3adafead8
[3] "Language Modeling with nn.Transformer and TorchText — PyTorch Tutorials 1.13.1+cu117 documentation." [Online]. Available: https://pytorch.org/tutorials/beginner/transformer_tutorial.html
[4] "Better Language Models and Their Implications," Feb. 2019. [Online]. Available: https://openai.com/blog/better-language-models/
[5] "Steganography Tutorial | A Complete Guide For Beginners," Jan. 2019. [Online]. Available: https://www.edureka.co/blog/steganography-tutorial
[6] R. S. R. Prasad and K. Alla, "A new approach to telugu text steganography," in *2011 IEEE Symposium on Wireless Technology and Applications (ISWTA)*. IEEE, 2011, pp. 60–65.
[7] K. Bennett, "Linguistic steganography: Survey, analysis, and robustness concerns for hiding information in text," 2004.
[8] A. M. Hamdan and A. Hamarsheh, "Ah4s: an algorithm of text in text steganography using the structure of omega network," *Security and Communication Networks*, vol. 9, no. 18, pp. 6004–6016, 2016.
[9] Z. Yang, N. Wei, Q. Liu, Y. Huang, and Y. Zhang, "Gan-tstega: Text steganography based on generative adversarial networks," in *Digital Forensics and Watermarking: 18th International Workshop, IWDW 2019, Chengdu, China, November 2–4, 2019, Revised Selected Papers 18*. Springer, 2020, pp. 18–31.
[10] H. Kang, H. Wu, and X. Zhang, "Generative text steganography based on lstm network and attention mechanism with keywords," *Electronic Imaging*, vol. 2020, no. 4, pp. 291–1, 2020.
[11] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6
[12] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
[13] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick, "Microsoft coco captions: Data collection and evaluation server," *arXiv preprint arXiv:1504.00325*, 2015.
[14] "Translation Task - ACL 2017 Second Conference on Machine Translation." [Online]. Available: https://www.statmt.org/wmt17/translation-task.html