Generative Models for Information Security

Student Name: John Jennings
Supervisor Name: Chris Willcocks

Submitted as part of the degree of MEng Computer Science to the
Board of Examiners in the Department of Computer Sciences, Durham University

Abstract —
Context/Background: Steganography, the practice of hiding information in plain sight, is a well researched topic within the information security sector, with many systems offering users the ability to encode secret information within an image or audio file. There has been little recent research, however, into lexical steganography, the task of embedding information within a piece of text.
Aims: This project investigates the application of cutting-edge natural language processing models to the task of lexical steganography, aiming to develop a method that takes an arbitrary English sentence as input, along with the secret payload, and returns a piece of text that is semantically equivalent to the input, indistinguishable from fluent English, and that can be decoded to reveal the payload.
Method: A number of novel methods were developed, including the training of a sequence-to-sequence adversarial autoencoder, the development of multiple interpolation schemes for the latent space of a pretrained sentence encoder, and the creation of a novel method for inducing paraphrasing behaviour in the GPT2 language model.
Results: An automatic evaluation of each method's embedding capacity was conducted and verified by a user study, measuring the relevance and fluency of each model's output. The evaluation results indicate that in their current state, the autoencoder-based models lack the necessary fluency to generate high-quality stegotexts. The GPT2 model, however, is shown to return highly fluent stegotexts at a capacity of four bits for 91.6% of tested samples, rivalling the performance of the current state-of-the-art system, CoverTweet.
Conclusions: In conclusion, we have found that high-capacity language models pretrained on multiple tasks and diverse datasets can generalise well to lexical steganography. In particular, we have found that a carefully constructed input can be effective in inducing paraphrasing behaviour in GPT2, from which a hash-based embedding scheme can assign payloads. In this study, we investigated several purpose-built architectures for lexical steganography. Our findings suggest that lexical steganography is primarily a problem of language modelling and that large models such as GPT2 generalise well in contrast to purpose-built architectures.

Keywords — Lexical steganography, paraphrase generation, language models

I INTRODUCTION

Steganography is the practice of embedding a secret message within an inconspicuous 'cover text'. It is often used in conjunction with cryptography to provide an additional layer of security, hiding not only the contents of the message, but the fact that a message has been sent at all. In recent years, this distinction has become increasingly important, with the growing use of metadata-based traffic analysis techniques that obtain information from the context surrounding a message, without knowledge of its contents. To quote the former General Counsel to the US

National Security Agency, Stewart Baker, "Metadata absolutely tells you everything about somebody's life. If you have enough metadata, you don't really need content" (Pugliese 2016).

This revelation has resulted in the proliferation of communication systems that are designed to obfuscate such metadata, most notably the Tor Project (Dingledine et al. 2004). With this motivation in mind, an effective steganography system poses a great threat to traffic analysis, as a message must first be identified before any metadata can be collected. With respect to more industrial applications, steganography is often discussed within the context of digital watermarking, the practice of embedding a secret identifier within a piece of data for the purposes of copyright protection (Cox et al. 2007).

While steganography research is a relatively active area, the vast majority of papers focus on embedding information within images or audio. This is to be expected as both forms of media have a high 'steganographic capacity', allowing for a large amount of information to be encoded through small, imperceptible changes to the cover data. A common example of this approach is encoding information within the least significant bits of an image or audio file (Cheddad et al. 2010, Gopalan 2003). Text, on the other hand, does not offer such a luxury. Flipping the least significant bit of an ASCII character is considerably more noticeable than flipping the same bit of a pixel.

Despite its lower capacity, the ubiquity of text data has motivated the development of numerous text-based steganography systems. Many of these systems, however, fall victim to a compromise between undetectability to humans and undetectability to machines. Systems that encode data through the use of special Unicode characters (Ali 2010) or modulating some property of the font (Xiang et al. 2007) may be largely imperceptible to humans, but can be detected by a single line of code. A more robust approach would be to encode information within the language itself, since a single idea can be expressed by a multitude of equally likely sentences. Known as 'lexical steganography', this is not a particularly new paradigm (Winstein 1999), although these approaches have until recently been hampered by a lack of suitable language models. With the recent surge in Natural Language Processing (NLP) performance, it would be expected that lexical steganography systems would improve in tandem, but it would appear that this area has been forgotten, with the latest major development published in 2014 (Wilson et al. 2014).

This paper aims to bridge the gap between cutting-edge NLP research and lexical steganography, developing a system that remains imperceptible to both humans and machines by drawing samples directly from the distribution of cover texts. Taking inspiration from the fields of neural machine translation and paraphrasing, we investigate the effectiveness of purpose-built sequence-to-sequence models and high capacity, general-purpose language models. Following a thorough evaluation of the fluency and relevance of each model's output, our results suggest that larger, general purpose language models can generalise well to the problem of lexical steganography, in comparison to the purpose-built approaches. The contributions of this paper include:

1. A novel method for conditioning GPT2 to perform the task of paraphrasing, achieving a state-of-the-art embedding capacity of 4 bits in 91.6% of tested samples.
2. The application of adversarial autoencoder, multi-task sentence encoder, and transformer-decoder models to the new problem of lexical steganography
3. The development of multiple single-point interpolation schemes for the latent space of a pre-trained sentence encoder
4. An evaluation of each model's performance, giving valuable insight into the geometry of the lexical steganography task

II RELATED WORK

A A Brief History of Lexical Steganography

Lexical steganography saw its inception with T-Lex (Winstein 1999). Using the WordNet synonym database (Miller 1995), T-Lex assigns a unique binary value to each word in a set of synonyms. Words in the cover text are then substituted with the appropriate synonyms, such that their binary values encode the payload data. While the system itself is shown to have a limited steganographic capacity, the paper provides a solid framework for lexical steganography in general, describing it as "finding degrees of freedom in some textual entity that allow an underlying work to be changed, while preserving its meaning". For T-Lex, these degrees of freedom were found in the choice of words, but the notion of hiding within the redundancies of natural language is the cornerstone from which all lexical steganography systems are built.

In the following years, lexical steganography received little research attention. A survey conducted in 2007 found that an average of only 3.9 papers were published per year in the period of 1999-2007 (Bergmair 2007), with many of these papers tackling the problem of cover generation, in which the content of the cover text is controlled entirely by the system (Chapman et al. 2001, Chand & Orgun 2006). While these systems do provide a useful method for concealing data, this paper will deal with the cover modification problem as described in Winstein 1999.

Of those papers that describe cover modification schemes, some propose improved synonym substitution systems, using more powerful language models, including parse trees (Topkara et al. 2006*a*), and refined versions of WordNet (Bolshakov & Gelbukh 2004, Topkara et al. 2006*b*), while others explore new areas of linguistic redundancy, such as presuppositions (Vybornova & Macq 2007), and spelling errors (Topkara et al. 2007). Unfortunately, the capacity of these systems was limited by the expressive power of available language models, with a recent survey finding that most systems had a capacity of less than one bit per sentence (Chang & Clark 2014).

The current cutting edge of lexical steganography is arguably the CoverTweet system (Wilson et al. 2014). With the ability to encode 4 bits into a 140 character tweet with a success rate of 40%, its capacity is considerably greater than previous systems. Drawing parallels to the problem of paraphrasing, a cover text is transformed into a number of candidate texts through repeated application of rules from the PPDB paraphrase database (Ganitkevitch et al. 2013). These candidate texts are then scored by a trigram language model trained on 75M tweets, and a secondary trigram model trained on previous tweets by the user.

The payload is embedded through the use of a keyed hash function (Fig. 1). First introduced by Grothoff et al. (2005) , this holds a number of advantages over traditional methods that use some linguistic element of the stegotext to represent a Huffman Code (Grothoff et al. 2005), or mixed-radix number (Winstein 1999). Using a hash function will improve the resilience of the system to steganalysis, since there will be little correlation between the content of the stegotext and the embedded payload, whereas in other systems, the presence of a particular word will always represent the same payload data. Additionally, a hash function is effective in decoupling the language model from the embedding scheme. While with other schemes, the recipient of a message will need access to the, often very large, codebook, with this scheme, the payload can be extracted simply by hashing the stegotext. The sender could even substitute the language model for a completely different architecture without needing to notify the recipient.

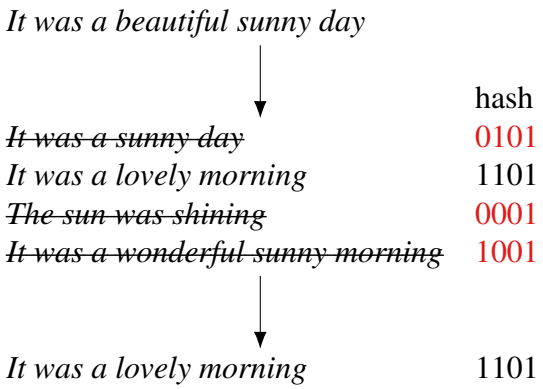


Figure 1: Embedding a message through the use of a hash function (Wilson et al. 2014)

Embedding via a hash function is not, however, without its downsides, requiring in the naïve case the evaluation of 2^n candidate texts on average to embed a payload of length n. Wilson et al. (2014) suggest a method of progressively hashing the text as it is generated, allowing for a suitable stegotext to be generated in linear time at the expense of an increased susceptibility to steganalysis.

B Natural Language Processing

While there have been no major developments in lexical steganography since CoverTweet, research in natural language processing has rapidly gained traction in recent years with the advent of recurrent neural networks (Goldberg 2016). Links between lexical steganography and areas of NLP such as machine translation and paraphrasing are long established (Grothoff et al. 2005, Bolshakov & Gelbukh 2004), as they share an underlying goal of understanding the meaning of an input text in order to generate a semantically equivalent output, whether it be expressed in a different language, or in a shorter number of words. It is therefore likely that new approaches in these fields could be used to improve the quality and capacity of existing stegosystems.

Recent developments in representation learning would be of particular use to lexical steganography, with embedding methods such as Word2Vec (Mikolov et al. 2013), GloVe (Pennington et al. 2014), and ELMo (Peters et al. 2018), transforming a set of words into a vector space modelling some notion of semantic similarity. Compared to simpler language models such as WordNet, word embeddings capture a more complex, generalized relationship, famously allowing for simple analogies through vector arithmetic, e.g. "king - man + woman = queen" (Vylomova et al. 2015). This expressive power could be used to increase the capacity of synonym substitution methods by replacing a word with one from a larger set of words that are not strictly synonymous, but still carry a similar meaning. This idea can be extended further with approaches such as the skip-thought model (Kiros et al. 2015), and Googles Universal Sentence Encoder (Alabish et al. 2013), which model the high-level meaning of entire sentences.

For lexical steganography, modelling the meaning of a text is only the beginning, the system must then generate a semantically similar output. For this purpose, the recurrent encoder-decoder models that dominate machine translation are likely to be of great use, extending the representation learning models discussed previously with a decoder that generates a new sentence from the semantic embedding of the input. Since their introduction, encoder-decoder models have improved drastically with the introduction of attention mechanisms that allow the decoder to focus

on different areas of the latent vector as the output is generated (Bahdanau et al. 2014), recently taken to the extreme with a family of encoders known as transformers that rely solely on attention to encode a sentence (Vaswani et al. 2017).

C Evaluation Methods

The effectiveness of a stegosystem can be defined as a combination of its capacity and detectability. That is, a strong stegosystem should embed as many bits in the cover text as possible, while ensuring that the generated stegotext cannot be identified as a product of the stegosystem. Capacity can be measured directly as the number of bits embedded in a cover text. Detectability, on the other hand, is a more abstract concept that must be quantified by a surrogate measure. The quality metrics used for Neural Machine Translation (NMT) are strong candidates for this role, since the quality of a translation system can be viewed as the ease at which one could distinguish a machine translation from a human-translated ground truth.

The Bilingual Evaluation Understudy (BLEU) score is the metric typically used for this purpose (Papineni et al. 2002), defined as

$$\text{BLEU} = \text{BP} \cdot \exp\Big(\sum_{n=1}^{N} w_n \log p_n\Big) \tag{1}$$

$$\text{BP} = \exp\Big(1 - \max\Big(1, \frac{r}{c}\Big)\Big) \tag{2}$$

where p_n is an n-gram precision score of the output text against a set of ground-truth references, w_n is an optional weighting for each n-gram score and BP is a 'brevity penalty' that reduces the score for a text of length c that is less than the length of the shortest reference r. The precision score is additionally modified to only count a token in the ground truth once, thereby penalising outputs that simply repeat correct tokens. As shown in Table 1, BLEU is generally a better indicator of textual similarity than naïve precision and the recall-based text summarisation metric, ROUGE (Lin 2004). However, previous work in evaluating natural language generation suggests that such metrics do not sufficiently capture the fluency of a text and should therefore only be used in conjunction with a final human evaluation (Cífka et al. 2018).

Table 1: Unigram quality measures for multiple texts against the reference texts "It was a beautiful day" and "It was a sunny morning"

Output	Precision	BLEU-1	ROUGE-1
it was a beautiful day	1.00	1.00	0.80
it was a sunny day	0.80	0.80	0.80
it was a sunny hotdog	0.80	0.80	0.80
it was a beautiful sunny morning day	0.71	0.71	1.00
was was was was was	1.00	0.20	0.20
it day	1.00	0.22	0.20

D Summary

In summary, we aim to build on previous approaches to lexical steganography, investigating the effectiveness of a hash-based embedding scheme applied to the current cutting-edge models used within natural language processing. We draw particular inspiration from the sequence-to-sequence models used within machine translation, and develop a number of methods for applying such models to lexical steganography. We additionally investigate a method of lexical steganography via a general-purpose language model conditioned to perform the task of paraphrase generation. These methods are evaluated with respect to their steganographic capacity and detectability in both a user study and an automatic evaluation using the BLEU score against a set of known paraphrase pairs.

III SOLUTION

The systems proposed in this paper embed a message into a cover text by repeatedly generating candidate stegotexts until the leading bits of the SHA256 hash match the intended payload. The crux of the problem is therefore in developing a conditional language model to generate a large number of high-quality paraphrases for a particular input. At a high level, this can be thought of as fitting a model, parameterized by θ, to maximise the conditional likelihood of each paraphrase $\mathbf{y}$ of a given cover text $\mathbf{x}$.

$$\theta^{MLE} = \underset{(\mathbf{x},\mathbf{y})\in D}{\operatorname{argmax}} \sum_{i=1}^{N} \log P(\mathbf{y}|\mathbf{x};\theta) \tag{3}$$

For this purpose, we investigate three models with differing levels of specialisation to the task of lexical steganography, namely a purpose-built adversarial autoencoder, a pre-trained sentence encoder used as a component within an autoencoder model, and the high-capacity general-purpose language model, GPT2.

A Adversarial Autoencoder

The paraphrasing task can be modelled as a problem of sequence-to-sequence representation learning, a popular approach in machine translation (Bahdanau et al. 2014). In such a model (Fig. 2), a recurrent neural network takes a variable-length sequence of tokens $\mathbf{x} = (x_1, \ldots, x_T)$ as input and within its hidden state, encodes the sequence into a fixed-length representation z, effectively learning a conditional distribution over a latent space $p_{enc}(z|x_1, \ldots, x_T)$. An autoregressive decoder then takes this latent vector and sequentially predicts each token in the output sequence $\mathbf{y} = (y_1, \ldots, y_{T'})$, modelling the distribution $p_{dec}(y_t|y_1, \ldots, y_{t-1}, z)$. This allows the model as a whole to learn a distribution $p(\mathbf{y}|\mathbf{x})$ in which $\mathbf{x}$ and $\mathbf{y}$ can be of different lengths.

This model is trained as an autoencoder with the objective of minimising the mean squared error between the word embeddings of the input and output sequences. Because the latent space z is of a significantly smaller dimension than the input space, the model is encouraged to encode the most salient features of the text in its latent embedding. Sampling around a point in the latent space should then ideally lead to decoded sentences that are semantically similar. This process is unfortunately not so simple, with previous work in applications of autoencoders to text finding

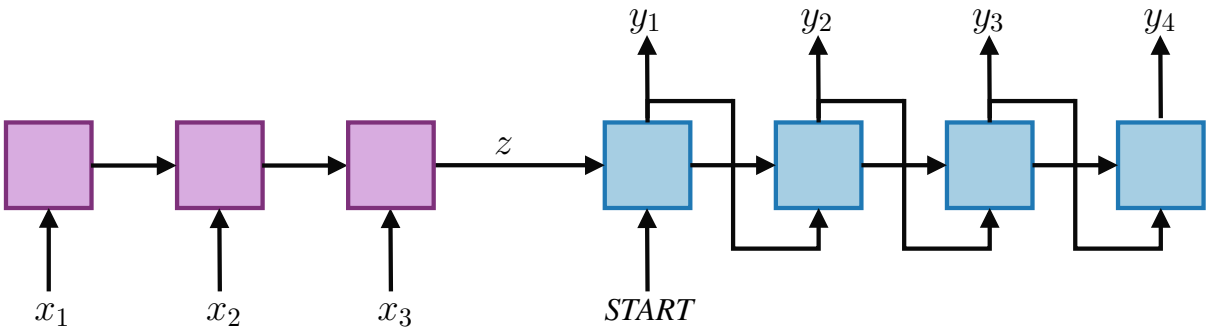


Figure 2: The sequence-to-sequence model (unrolled)

that due to its discrete nature, the learned embedding space is rarely smooth and interpolation often results in poor-quality sentences (Bowman et al. 2015).

To combat this issue, the final model (Fig. 3) is modified to act as a variational autoencoder (Kingma & Welling 2013), replacing the deterministic encoder with one that specifies the parameters of a diagonal gaussian distribution over z, encouraging the encoder to cover larger, smoother regions of the latent space. The posterior distribution is additionally constrained to follow a unit gaussian, preventing the encoder from simply encoding 'discrete' points as distributions with a very small variance. This is achieved through adversarial training with the inclusion of a Wasserstein critic trained to estimate the probability $p(g)$ that a given z is a sample from the encoder rather than a unit gaussian (Arjovsky et al. 2017). While this constraint would typically be enforced by the inclusion of a loss term using the KL divergence, recent research argues that adversarial training can offer stronger gradient signals (Makhzani et al. 2015) and indeed, a recent survey of autoencoder-based text generation models found that samples from an adversarial autoencoder were typically more semantically similar to their input than the corresponding samples from a VAE (Cífka et al. 2018).

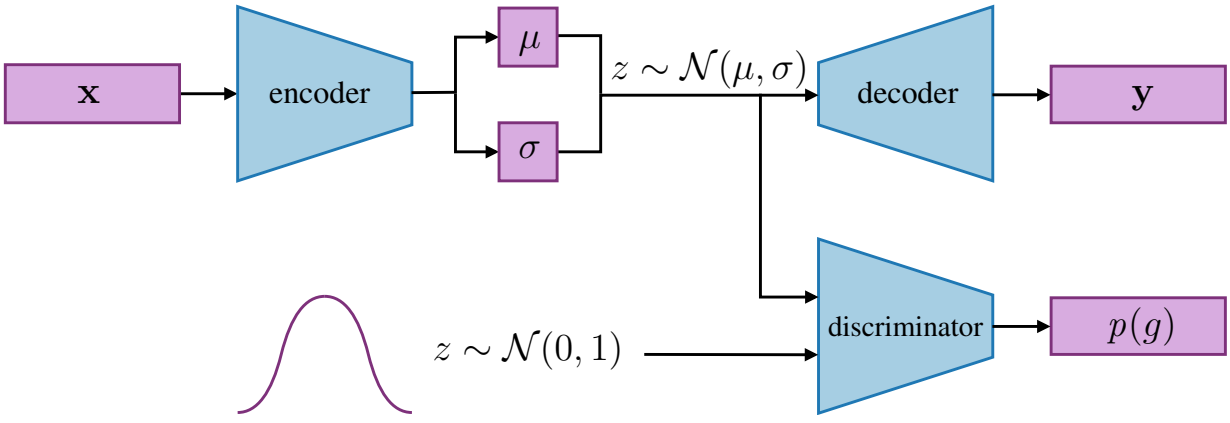


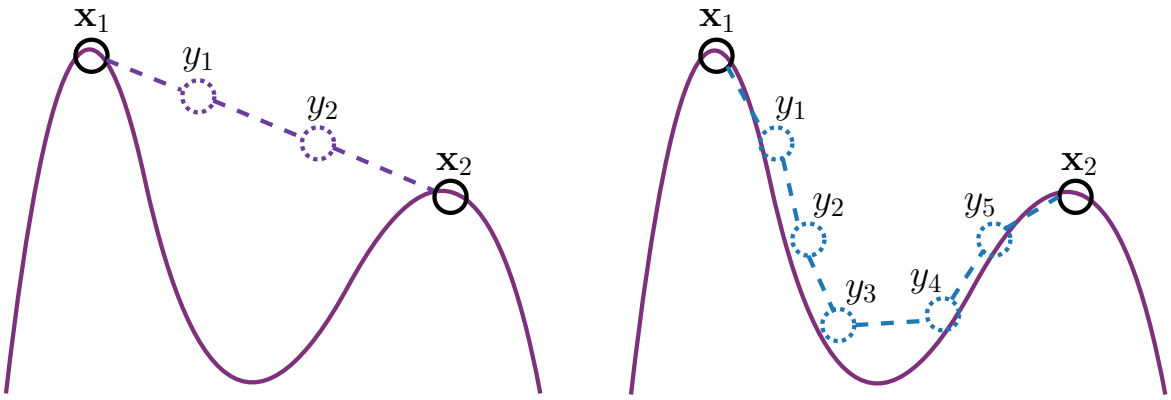
Figure 3: The Adversarial Autoencoder model

B Transfer Learning Autoencoder

A potential flaw with the AAE model is that the reconstruction task is used as a surrogate for the true goal of finding a semantic embedding for text. While learning a semantic embedding would indeed be a sensible solution to the task, it is not guaranteed that this will be the solution that the model converges to. This issue can be resolved by making use of the ubiquity of sequence-to-sequence models in NLP to learn a more robust embedding via multi-task learning. Subramanian et al. (2018) provide such a model, training a single encoder against multiple decoders, each given a unique task from machine translation, to natural language inference and constituency parsing.

To repurpose this model for lexical steganography, the pre-trained sentence encoder can be used to train a decoder on the reconstruction task, with the encoder weights frozen in order to preserve their high-quality embeddings. Unlike the AAE model, this encoder is deterministic and the shape of the latent space is unknown. This means that extra care must be taken to ensure that interpolated points remain on the manifold. Subramanian et al. (2018) propose an approach for interpolating between two sentences $\mathbf{x}_1$ and $\mathbf{x}_2$ using the gradient signals from the decoder (Fig. 4). Starting from the latent embedding of $\mathbf{x}_1$, the current position in the latent space h is updated in fixed steps of length α in accordance to Equation 4, and then passed to the decoder to generate a new interpolated sentence q.

$$h' = h + \alpha \nabla_h \log P(\mathbf{x}_2|h) \tag{4}$$



(a) (b)

Figure 4: Illustrations of linear interpolation (a) and gradient-based interpolation (b)

B.1 Multi-Path Interpolation

In order to interpolate around the embedding of a given text t, we explore two modifications to the scheme provided. The Multi-Path approach (Algorithm 1) simply chooses a random point on the manifold r and performs gradient-based interpolation between r and t, repeating the process until enough unique texts are found. To ensure that all returned texts are suitably similar to p, a decoded text is only accepted if its BLEU score with relation to p is greater than 0.2.

Algorithm 1: Multi-Path Interpolation (Forward)

Data: t, n, α
Result: Set Q consisting of n paraphrases of the text t

$\mathrm{Q} \leftarrow \varnothing$;
$r \leftarrow$ a randomly selected point on the manifold;
$h \leftarrow \mathrm{encode}(r)$;
while $|\mathrm{Q}| > n$ **do**
  $h \leftarrow h + \alpha \nabla_h \log P(t|h)$;
  $q \leftarrow \mathrm{decode}(h)$;
  if $\mathrm{BLEU}(q, t) < 0.2$ **then**
    $\mathrm{Q} \leftarrow \mathrm{Q} \cup \{q\}$;
    if $q == t$ **then**
      $r \leftarrow$ a randomly selected point on the manifold;
      $h \leftarrow \mathrm{encode}(r)$;
    end
  end
end
return Q;

B.2 Wandering Interpolation

A possible issue with the multi-path approach is that depending on the randomly selected point, the algorithm may spend a large amount of time in distant areas of the manifold that are unlikely to decode to similar texts. The Wandering Interpolation scheme (Algorithm 2) aims to solve this issue by constraining the exploration to latent vectors close to t. This is achieved by starting from t and at each step, choosing a new target r that is equal to t with probabilty p and otherwise, a randomly chosen point on the manifold. The value of p can then be manually tuned to reach a suitable balance between exploring the latent space and staying in the vicinity of t.

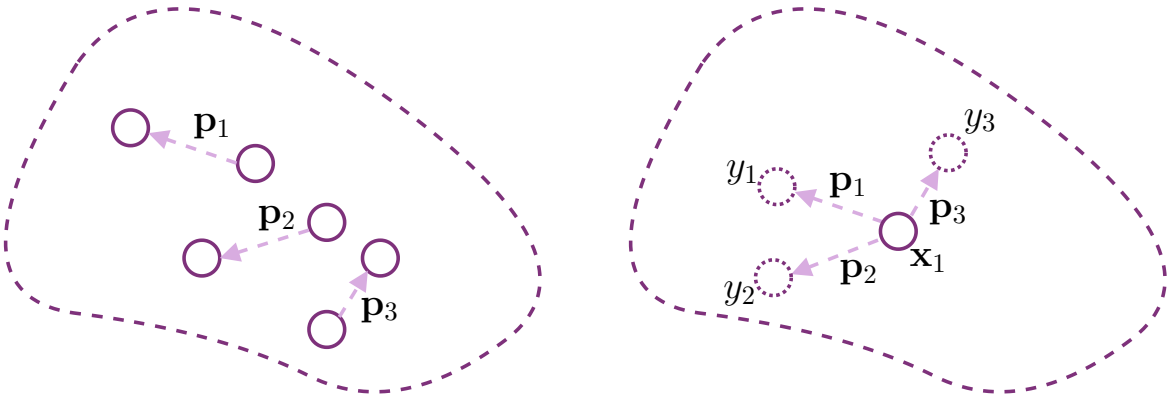
Algorithm 2: Wandering Interpolation

Data: t, n, α, p
Result: Set Q consisting of n paraphrases of the text t

$\mathrm{Q} \leftarrow \varnothing$;
$h \leftarrow \mathrm{encode}(t)$;
while $|\mathrm{Q}| > n$ **do**
  $r \leftarrow t$ with probability p, else a randomly selected point on the manifold;
  $h \leftarrow h + \alpha \nabla_h \log P(r|h)$;
  $q \leftarrow \mathrm{decode}(h)$;
  if $\mathrm{BLEU}(q, t) < 0.2$ **then**
    $\mathrm{Q} \leftarrow \mathrm{Q} \cup \{q\}$;
  end
end
return Q;

B.3 Analogical Interpolation

An alternative method can be devised for analogical interpolation by performing vector arithmetic on a set of known paraphrase pairs. Previous work has shown that for word embeddings, a surprisingly robust set of analogies of the form "x is to y as z is to ?" can be computed through simple combinations of vectors (Vylomova et al. 2015). For instance, taking the vector difference of the word embeddings for 'paris' and 'france' has been shown to yield a vector relating to the concept of a capital city, such that when added to the embedding of 'warsaw', a vector close to the embedding of 'poland' is obtained. Following the same process, a small set of sentences and their associated paraphrases can be used to create a set of vectors $\mathbf{p}$ in the latent space that correspond to the concept of paraphrasing. Interpolations around a particular sentence can then be generated by applying these paraphrase vectors to the embedded sentence (Fig. 5) and decoding the output.



(a) (b)

Figure 5: Illustration of the analogical interpolation scheme

C Conditioning GPT2

To extend the idea of repurposing existing models further, we investigate a method for conditioning the GPT2 model (Radford et al. 2019), such that it is encouraged to output paraphrases of a given sentence. GPT2 differs substantially from the previously discussed models, using a powerful multi-headed self-attention mechanism to capture the internal dependencies of the input sequence. The encoder-decoder architecture is replaced by a decoder-transformer (Fig. 6) which models the input $\mathbf{x}$ and output $\mathbf{y}$ as a single combined sentence $\mathbf{m} = (x_1, \ldots, x_T, \delta, y_1, \ldots, y_{T'})$ where δ is a special delimiter character (Liu et al. 2018). The model is trained to maximise the likelihood of these combined sentences with the standard objective function, modified to operate within a context window of length k.

$$\mathcal{L} = \sum_i \log P(m_i | m_{i-k}, \ldots, m_{i-1}; \Theta) \tag{5}$$

This ability to in some sense encode the problem definition within the input allows for the model to be easily trained on a wide range of tasks with minimal additional resources required, in comparison to the multi-task sequence-to-sequence model that required a separate decoder for

each task. Along with the higher levels of parallelisation available from its lack of recurrent elements, this allows the model to be scaled up significantly, with Radford et al. training a model with 1.5B parameters on a corpus of text measuring 40GB. GPT2 has been shown to generalise spectacularly well to tasks within NLP, with a zero-shot performance on the question answering task exceeding that of 3 out of 4 baseline models (Radford et al. 2019).

We therefore propose a scheme for constructing an input to GPT2 that implicitly encodes the task of paraphrasing. As shown in Fig. 7, the model is supplied with a sequence of carefully delimited paraphrase pairs $\mathbf{m}_{cond}$ followed by $\mathbf{x}$. If GPT2 has in fact generalised well to language modelling then it will infer from this input pattern that a likely continuation of the input would be a paraphrase of $\mathbf{x}$. Additional paraphrases can then be generated by conditioning on an $\mathbf{m}_{cond}$ consisting of different paraphrase pairs.

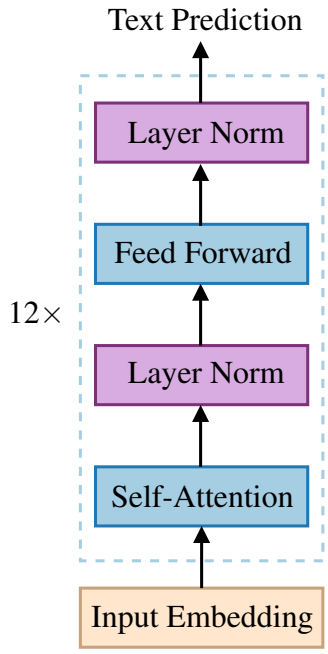


Figure 6: The GPT2 model

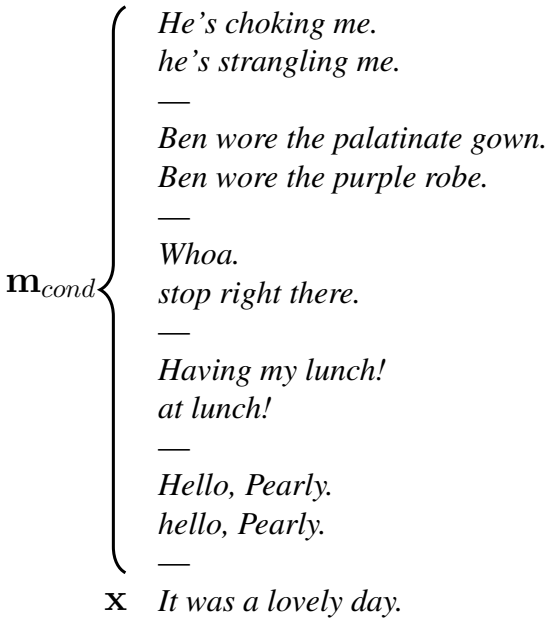


Figure 7: Example input for paraphrasing "It was a lovely day."

IV RESULTS

Evaluation was performed using the PARANMT corpus, a set of 50M paraphrase pairs with an associated 'paraphrase score' to denote the extent of semantic equivalence between each pair (Wieting & Gimpel 2017). The paraphrase pairs were generated using a corpus of translations from English to Czech, with each pair consisting of the original English sentence and the associated Czech sentence translated into English via an NMT model.

A Capacity

We first investigate the steganographic capacity of each approach. This is achieved by plotting the textual similarity between each system's output and the 10,000 highest scoring paraphrase pairs in PARANMT, with the understanding that a system's steganographic capacity is the highest

number of encoded payload bits at which the returned stegotexts are both fluent and semantically similar to the cover text. Since fluency cannot be automatically quantified, our estimations are based only on the relevance of the stegotext with respect to the cover text, measured as the BLEU-2 score between the stegotext and both the original sentence and its paraphrase in PARANMT, with fluency later measured in a user study. Relevance measures are taken for each model, set to encode random payloads of lengths in the range of 1 to 4 bits.

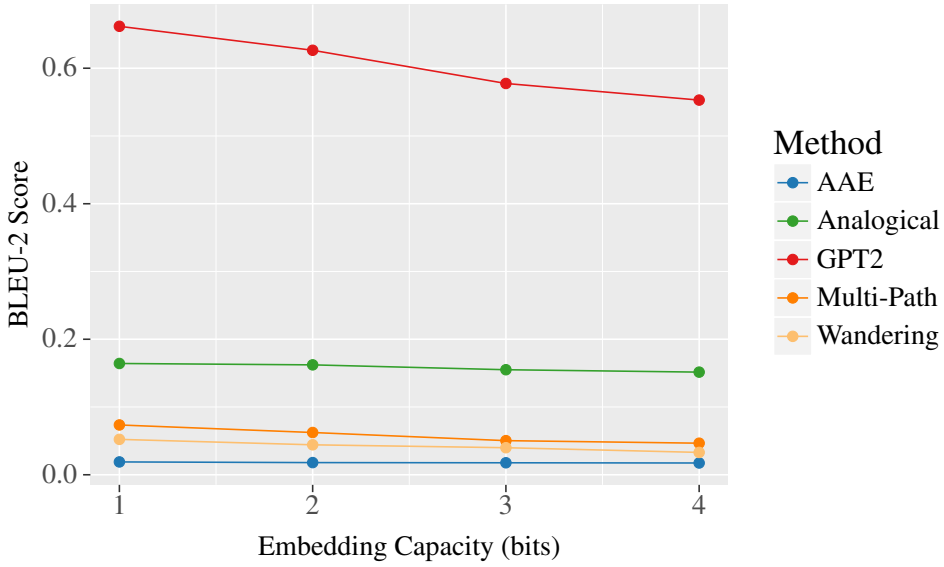


Figure 8: Mean output relevance over increasing embedding capacities

As shown in Fig. 8, the AAE model does not appear to generalise to unseen samples, receiving a very low BLEU score even for a capacity of one. Results are marginally improved in the transfer learning model, with the analogical interpolation scheme appearing to outperform the gradient-based approaches, although a BLEU score in the range of 0.1 to 0.2 is still relatively poor. Performance for GPT2 however, is noticeably better across the board, achieving BLEU scores of around 0.6 with only a minor deterioration as the capacity is increased. Providing the output for GPT2 is also fluent, this would indicate a capacity rivalling that of CoverTweet, the current leading lexical steganography system.

B User Study

In order to verify the results of the metric-based evaluation, and further investigate the quality of each system's output, a user study was performed using the method described by Cífka et al. (2018). Twenty participants were supplied with a set of 25 samples from each model, set to encode with a capacity of four bits, along with the associated paraphrase sample from PARANMT. The participants were then instructed to score each sample on a 1-5 Likert scale for both the relevance of the output to the original sentence, and the overall fluency of the text, regardless of meaning. For the transfer learning model, we used the analogical interpolation scheme as it was the best performing method in the automatic evaluation.

B.1 Relevance

As shown in Fig. 9, the user evaluations for relevance support the initial metric-based evaluation, with the AAE model scoring very poorly, the transfer learning model offering a slight improvement, and GPT2 vastly outperforming both models with a mean relevance score of 3.93, compared to the mean PARANMT score of 4.64. A set of paired Student's t-tests were performed on these results (Table 2), indicating with a high level of confidence that the samples from GPT2 are of a higher relevance on average than both the AAE and transfer learning models, but less relevant than samples from PARANMT.

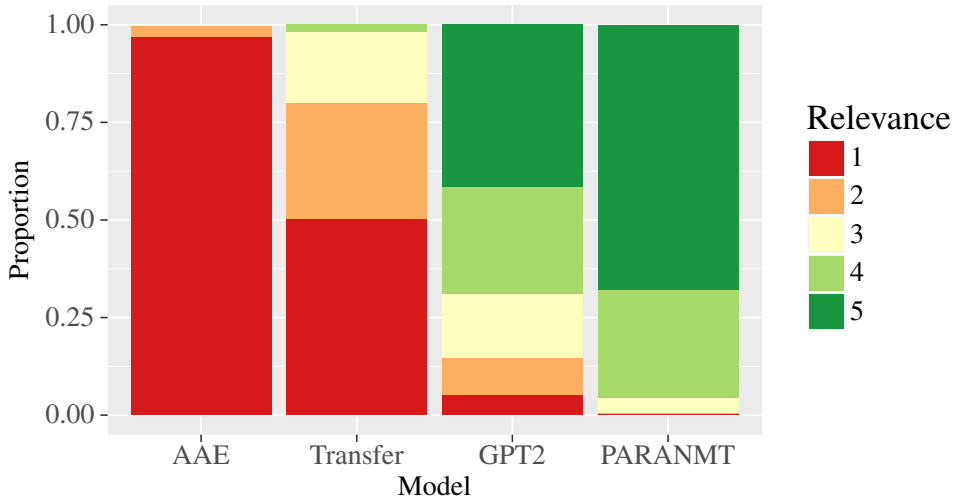


Figure 9: Spread of user-evaluated relevance scores

Table 2: User Study Results - Relevance

Source	Mean Score	σ	T Score			
			AAE	Transfer	GPT2	PARANMT
AAE	1.030	0.081	-	7.556**	39.442**	68.193**
Transfer	1.784	0.468	-	-	36.649**	35.667**
GPT2	3.932	0.343	-	-	-	14.994**
Ground Truth	4.635	0.241	-	-	-	-

Note: $*p < .05$, $**p < .01$

B.2 Fluency

As shown in Fig. 10, the user evaluation scores for fluency showed a similar pattern to those for relevance. The AAE model performed poorly, with over 75% of samples assigned a fluency score of one (Gibberish), and similarly for the transfer learning model, receiving a score of one for slightly under half of its samples. The output fluency of the GPT2 model was considerably better, with almost 75% of samples given a score of five (Convincing English). Significance testing again rejected the null hypothesis in all cases (Table 3), indicating with a high level of confidence that the difference in mean scores was not due to random chance.

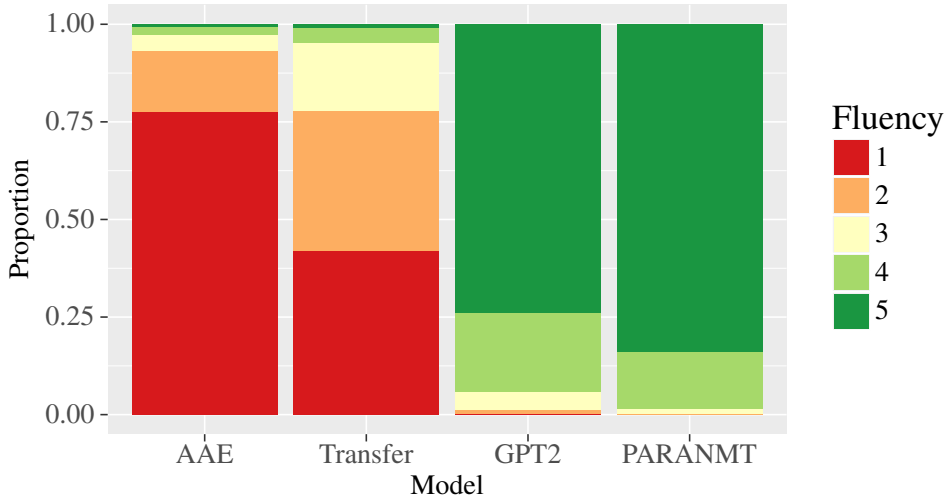


Figure 10: Spread of user-evaluated fluency scores

Table 3: User Study Results - Fluency

Source	Mean Score	σ	T Score			
			AAE	Transfer	GPT2	PARANMT
AAE	1.325	0.427	-	5.954**	36.203**	36.159**
Transfer	1.861	0.547	-	-	27.072**	24.609**
GPT2	4.668	0.225	-	-	-	4.212**
Ground Truth	4.820	0.219	-	-	-	-

Note: $*p < .05$, $**p < .01$

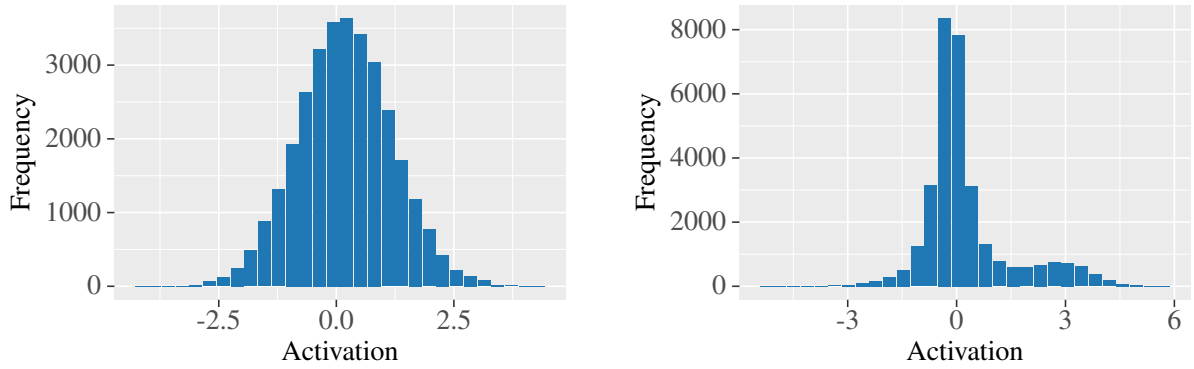
V EVALUATION

With regards to the aim of this paper, investigating the applications of cutting-edge NLP models to lexical steganography, we now discuss the strengths and limitations of each developed method, and their implications for the development of future systems.

A Adversarial Autoencoder

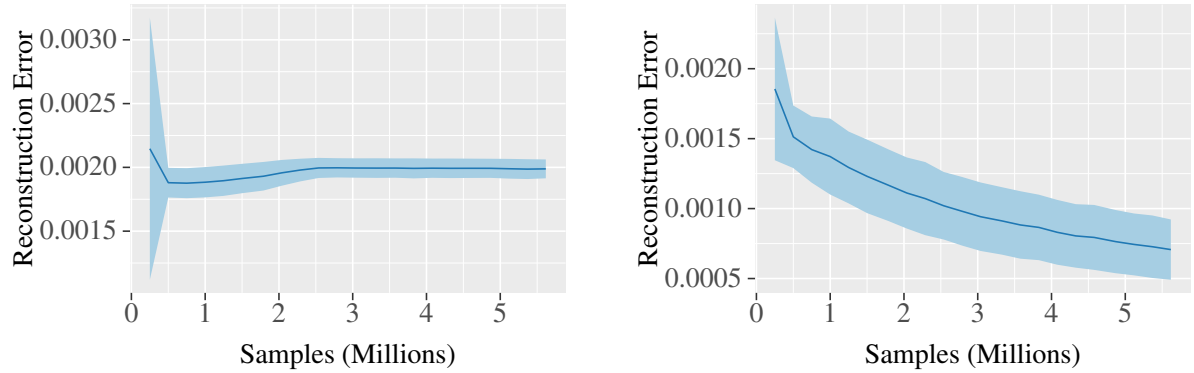
By all measures, the adversarial autoencoder was not successful in its task, failing to encode even a single bit into the majority of tested samples. It may simply be the case that such a model is not suitable for this application, with Subramanian et al. (2018) suggesting that the strong restrictions on the parametric form of the posterior may hamper the model's ability to encode information in the latent vector. While it may be possible to learn such a mapping for MNIST, attempting to do so for a distribution as discrete and high-dimensional as the English language is a problem so difficult that it takes priority over the reconstruction objective.

This problem quickly became evident during training as the model appeared to be highly sensitive to changes in α, the weight of the posterior regularisation term . As shown in Figures 11 and 12, when the loss terms were equally weighted, the model experienced 'posterior collapse'



(a) (b)

Figure 11: Aggregate posteriors for the AAE model trained at $\alpha = 1$ (a) and 0.0001 (b)



(a) (b)

Figure 12: Reconstruction error (training) for the AAE model trained at $\alpha = 1$ (a) and 0.0001 (b)

(Bowman et al. 2015), forcing the latent encoding to follow the correct distribution at the expense of ignoring the reconstruction term completely. Lowering the weight resulted in a sharp transition to the opposite problem, with there appearing to be no point at which the model would optimise both objectives in tandem. The behaviour of the model in its current form additionally suggests that the reconstruction objective does not lend itself well to paraphrase generation. Sampling around a particular point would often result in the same 'base sentence' varied around its most uncommon word, usually changing the meaning of the sentence significantly.

B Transfer Learning Autoencoder

While the transfer learning model did appear to generalise more easily to unseen data than the adversarial autoencoder, the gradient-based interpolation schemes struggled to generate a large number of fluent paraphrases. This suggests that paraphrases do not necessarily occupy the same local area, and thus the gradient-based approaches are unlikely to find them. This idea is further supported by the relative success of the analogical interpretation scheme, which does not limit itself to any particular area of the latent space. The analogical method does however make the assumption that the latent space has a set of globally consistent 'paraphrase vectors'. The partial success of the analogical method suggests that this assumption is true to a certain extent, but it is

more likely that the paraphrase vectors point to promising regions of the latent space, rather than exact points. Indeed, in the original investigation into word embeddings, 'king - man + woman' pointed to a vector close to queen, but not necessarily queen itself (Mikolov et al. 2013).

C GPT2

Sampling from GPT2 proved to be a highly succesful method for stegotext generation, successfully embedding a 4-bit message into 91.6% of samples, compared to a success rate of only 40% for the CoverTweet system. Additionally, the GPT2 method does not restrict itself to the domain of Twitter, with the user study including samples from a wide variety of domains such as EU legislation, scientific writing and film scripts. This arguably state-of-the-art capacity was achieved without sacrificing the undetectability of the embedding, with over 75% of tested samples marked as being indistinguishable from fluent English, and only marginally less fluent than the best samples of PARANMT, which was effectively embedding at a zero-bit capacity.

As for why GPT2, a model designed for language modelling, would outperform models specifically designed with lexical steganography in mind, we must first take into account that GPT2 is of a scale orders of magnitude larger, trained on a set of 8M documents, using 32 Google TPUv3's (roughly $2048 per hour) (Radford et al. 2019). The AAE and transfer learning models were trained with a single Titan X card using a dataset of 1.5M and 60M sentences respectively. It would be natural to assume that their performance would be improved if they were trained at the same scale as GPT2.

It would be naïve, however, to assume that size is the only contributing factor to the huge difference in performance. The language modelling training objective can be seen as placing a greater importance on fluency, whereas the reconstruction objective focusses more on relevance. While these two factors may be equally important for paraphrase generation, it could be argued that for lexical steganography, fluency is necessary to prevent detection while a slight deviation in relevance is tolerable. For the AAE model in particular, it was clear from the user study that the output fluency was so poor that it was often difficult to extract any meaning to evaluate for relevance. As part of its pre-training, GPT2 was additionally trained on the task of semantic similarity classification, in which the model is tasked with determining if two given sentences express the same meaning. Pre-training the model on the task of identifying paraphrases is likely to have encouraged the model to learn internal representations that would be of great use for paraphrase generation.

VI FUTURE WORK

This project leaves a number of areas for which future work could expand upon. For the adversarial autoencoder, training on a larger, more varied corpus is likely to improve results. Additional work could investigate the performance of adversarial autoencoders where the posterior is constrained to follow a distribution other than a diagonal Gaussian, potentially resolving the posterior collapse problem. Within the interpolation schemes developed for the transfer learning autoencoder, new hybrid methods could take the paraphrase vectors generated by the analogical scheme and apply the gradient-based methods to explore these promising regions, or interpolate towards them.

With regards to GPT2, our method conditions the pre-trained model as it is given by Radford et al. (2019). Further success could be found in fine-tuning the model on a task formulated

specifically for lexical steganography. This could simply be the reconstruction task, with the aim of retaining the impressive fluency of the pre-trained model while steering it to behave in a similar fashion to an autoencoder. Alternatively, the model could be fine-tuned explicitly on the task of paraphrase generation, using corpora such as PARANMT to train GPT2 to maximise the likelihood of known paraphrases of a given sentence, using the stochasticity within the beam search decoder to generate alternative samples.

The focus of this project has been squarely on the language modelling aspects of lexical steganography, simply using SHA-256 as the embedding scheme. While embedding via a hash function is beneficial to the system's capacity, ensuring that there are minimal collisions between candidate stegotexts, this method does require the brute-force generation of multiple full stegotexts for each payload, an inefficiency that is a particular issue when encoding a payload on a system lacking a GPU. Further work could investigate the use of progressive hash functions, as used by Wilson & Ker (2016), allowing for the encoding process to be pre-empted if the eventual stegotext will not hash to the desired payload.

Finally, due to the lack of availability of both the CoverTweet model and the dataset it was evaluated on, we were unable to provide a direct comparison between it and models developed in this project, preventing the GPT2 method from making a definite claim to a state-of-the-art capacity. Within our evaluation of a model's detectability, we show each model's resilience to human detection but do not evaluate it against automatic detection methods. Further work could investigate the effectiveness of such methods, ranging from simple statistical analysis to training a classifier on the model's output.

VII CONCLUSION

In conclusion, our investigation has found that high-capacity language models pretrained on multiple tasks and diverse datasets can generalise well to lexical steganography. In particular, we have found that a carefully constructed input can be effective in inducing paraphrasing behaviour in GPT2, from which a hash-based embedding scheme can assign payloads. Our investigation into purpose-built lexical steganography models has found that language modelling plays a significant role in the underlying problem, and that models designed to prioritise fluency over relevance are likely to yield an increased performance in both measures.

We additionally propose several novel interpolation schemes for the latent space of a pretrained sentence encoder, the evaluations of which give a great deal of insight into the geometry of the problem, suggesting that semantically similar sentences do not necessarily occupy the same local area and that to a certain extent, vector-based analogy can be used to guide the interpolation towards promising regions of the latent space.

References

Alabish, A., Goweder, A. & Enakoa, A. (2013), 'A universal lexical steganography technique', *International Journal of Computer and Communication Engineering* **2**(2), 153.

Ali, A. E. (2010), 'A new text steganography method by using non-printing unicode characters', *Engineering and Technology Journal* **28**(1), 72–83.

Arjovsky, M., Chintala, S. & Bottou, L. (2017), 'Wasserstein gan', *arXiv preprint arXiv:1701.07875* .

Bahdanau, D., Cho, K. & Bengio, Y. (2014), 'Neural machine translation by jointly learning to align and translate', *arXiv preprint arXiv:1409.0473* .

Bergmair, R. (2007), A comprehensive bibliography of linguistic steganography, *in* 'Security, Steganography, and Watermarking of Multimedia Contents IX', Vol. 6505, International Society for Optics and Photonics, p. 65050W.

Bolshakov, I. A. & Gelbukh, A. (2004), Synonymous paraphrasing using wordnet and internet, *in* 'International Conference on Application of Natural Language to Information Systems', Springer, pp. 312–323.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R. & Bengio, S. (2015), 'Generating sentences from a continuous space', *arXiv preprint arXiv:1511.06349* .

Chand, V. & Orgun, C. O. (2006), Exploiting linguistic features in lexical steganography: design and proof-of-concept implementation, *in* 'System Sciences, 2006. HICSS'06. Proceedings of the 39th Annual Hawaii International Conference on', Vol. 6, IEEE, pp. 126b–126b.

Chang, C.-Y. & Clark, S. (2014), 'Practical linguistic steganography using contextual synonym substitution and a novel vertex coding method', *Computational Linguistics* **40**(2), 403–448.

Chapman, M., Davida, G. I. & Rennhard, M. (2001), A practical and effective approach to large-scale automated linguistic steganography, *in* 'International Conference on Information Security', Springer, pp. 156–165.

Cheddad, A., Condell, J., Curran, K. & Mc Kevitt, P. (2010), 'Digital image steganography: Survey and analysis of current methods', *Signal processing* **90**(3), 727–752.

Cífka, O., Severyn, A., Alfonseca, E. & Filippova, K. (2018), 'Eval all, trust a few, do wrong to none: Comparing sentence generation models', *arXiv preprint arXiv:1804.07972* .

Cox, I., Miller, M., Bloom, J., Fridrich, J. & Kalker, T. (2007), *Digital watermarking and steganography*, Morgan Kaufmann.

Dingledine, R., Mathewson, N. & Syverson, P. (2004), Tor: The second-generation onion router, *in* 'Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13', SSYM'04, USENIX Association, Berkeley, CA, USA, pp. 21–21.
URL: *http://dl.acm.org/citation.cfm?id=1251375.1251396*

Ganitkevitch, J., Van Durme, B. & Callison-Burch, C. (2013), Ppdb: The paraphrase database, *in* 'Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies', pp. 758–764.

Goldberg, Y. (2016), 'A primer on neural network models for natural language processing', *Journal of Artificial Intelligence Research* **57**, 345–420.

Gopalan, K. (2003), Audio steganography using bit modification, *in* 'Multimedia and Expo, 2003. ICME'03. Proceedings. 2003 International Conference on', Vol. 1, IEEE, pp. I–629.

Grothoff, C., Grothoff, K., Alkhutova, L., Stutsman, R. & Atallah, M. (2005), Translation-based steganography, *in* 'International Workshop on Information Hiding', Springer, pp. 219–233.

Kingma, D. P. & Welling, M. (2013), 'Auto-encoding variational bayes', *arXiv preprint arXiv:1312.6114* .

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A. & Fidler, S. (2015), Skip-thought vectors, *in* 'Advances in neural information processing systems', pp. 3294–3302.

Lin, C.-Y. (2004), 'Rouge: A package for automatic evaluation of summaries', *Text Summarization Branches Out* .

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L. & Shazeer, N. (2018), 'Generating wikipedia by summarizing long sequences', *arXiv preprint arXiv:1801.10198* .

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I. & Frey, B. (2015), 'Adversarial autoencoders', *arXiv preprint arXiv:1511.05644* .

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013), 'Efficient estimation of word representations in vector space', *arXiv preprint arXiv:1301.3781* .

Miller, G. A. (1995), 'Wordnet: A lexical database for english', *Commun. ACM* **38**(11), 39–41. **URL:** *http://doi.acm.org/10.1145/219717.219748*

Papineni, K., Roukos, S., Ward, T. & Zhu, W.-J. (2002), Bleu: a method for automatic evaluation of machine translation, *in* 'Proceedings of the 40th annual meeting on association for computational linguistics', Association for Computational Linguistics, pp. 311–318.

Pennington, J., Socher, R. & Manning, C. (2014), Glove: Global vectors for word representation, *in* 'Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)', pp. 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. & Zettlemoyer, L. (2018), 'Deep contextualized word representations', *arXiv preprint arXiv:1802.05365* .

Pugliese, J. (2016), *Death by metadata: the bioinformationalisation of life and the transliteration of algorithms to flesh*, Palgrave Macmillan, United Kingdom, pp. 3–20.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. & Sutskever, I. (2019), 'Language models are unsupervised multitask learners'.

Subramanian, S., Mudumba, S. R., Sordoni, A., Trischler, A., Courville, A. C. & Pal, C. (2018), Towards text generation with adversarially learned neural outlines, *in* 'Advances in Neural Information Processing Systems', pp. 7551–7563.

Subramanian, S., Trischler, A., Bengio, Y. & Pal, C. J. (2018), 'Learning general purpose distributed sentence representations via large scale multi-task learning', *CoRR* **abs/1804.00079**.

Topkara, M., Topkara, U. & Atallah, M. J. (2006*a*), Words are not enough: sentence level natural language watermarking, *in* 'Proceedings of the 4th ACM international workshop on Contents protection and security', ACM, pp. 37–46.

Topkara, M., Topkara, U. & Atallah, M. J. (2007), Information hiding through errors: a confusing approach, *in* 'Security, Steganography, and Watermarking of Multimedia Contents IX', Vol. 6505, International Society for Optics and Photonics, p. 65050V.

Topkara, U., Topkara, M. & Atallah, M. J. (2006*b*), The hiding virtues of ambiguity: Quantifiably resilient watermarking of natural language text through synonym substitutions, *in* 'Proceedings of the 8th Workshop on Multimedia and Security', MM&Sec '06, ACM, New York, NY, USA, pp. 164–174.
URL: *http://doi.acm.org/10.1145/1161366.1161397*

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. (2017), Attention is all you need, *in* 'Advances in Neural Information Processing Systems', pp. 5998–6008.

Vybornova, O. & Macq, B. (2007), A method of text watermarking using presuppositions, *in* 'Security, Steganography, and Watermarking of Multimedia Contents IX', Vol. 6505, International Society for Optics and Photonics, p. 65051R.

Vylomova, E., Rimell, L., Cohn, T. & Baldwin, T. (2015), 'Take and took, gaggle and goose, book and read: Evaluating the utility of vector differences for lexical relation learning', *arXiv preprint arXiv:1509.01692* .

Wieting, J. & Gimpel, K. (2017), 'Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations', *arXiv preprint arXiv:1711.05732* .

Wilson, A., Blunsom, P. & Ker, A. D. (2014), Linguistic steganography on twitter: hierarchical language modeling with manual interaction, *in* 'Media Watermarking, Security, and Forensics 2014', Vol. 9028, International Society for Optics and Photonics, p. 902803.

Wilson, A. & Ker, A. D. (2016), 'Avoiding detection on twitter: embedding strategies for linguistic steganography', *Electronic Imaging* **2016**(8), 1–9.

Winstein, K. (1999), Lexical steganography through adaptive modulation of the word choice hash, *in* 'Secondary education at the Illinois Mathematics and Science Academy'.

Xiang, L., Sun, X., Luo, G. & Gan, C. (2007), Research on steganalysis for text steganography based on font format, *in* 'Information Assurance and Security, 2007. IAS 2007. Third International Symposium on', IEEE, pp. 490–495.