Generative Models for Information Security: Literature Review

Student: John Jennings **Supervisor:** Chris Willcocks

September 28, 2018

Problem Definition

Generative models are important tools in machine learning, seeing recent success in a wide variety of applications, from synthesizing realistic images of human faces and house interiors, to transforming photos from summer to winter and night to day. Within the information security sector, they have been used for learning novel encryption schemes, modelling password distributions, and steganography techniques for hiding information within an image. This project will expand upon current research, and investigate the applications of generative models to lexical steganography.

Definition of Terms

Steganography - The art of concealed communication, differing from cryptography in that the very existence of the message is secret. This is achieved by embedding the message into a seemingly innocuous **cover text** that can be sent to the intended recipient without arousing suspicion.

Stegotext - The result of steganographically embedding a message into a particular cover text.

Lexical Steganography - A form of steganography concerned with embedding a message within written text through the use of redundancies in natural language e.g. word ordering, inclusion of adjectives or choice of synonyms.

Generative Model - A model that learns the underlying distribution of the training data in order to generate new samples from the same distribution.

Identified Issues and Themes

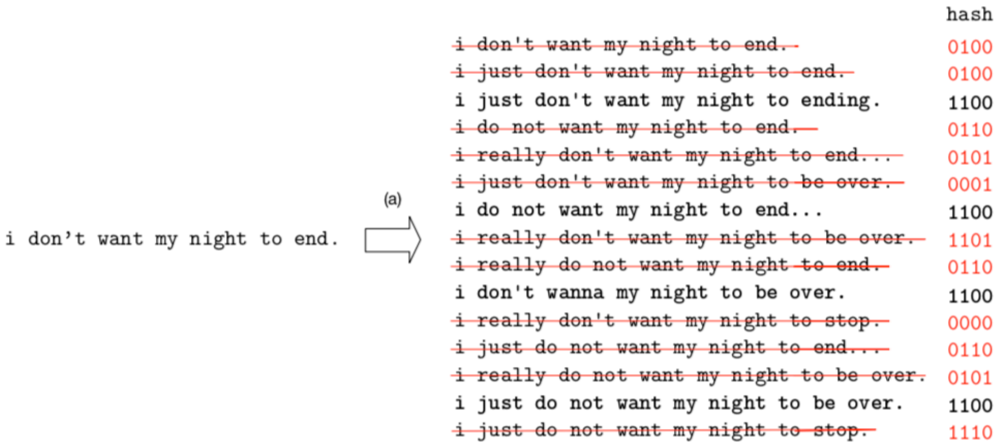
Correspondence to Neural Machine Translation

While the concept of lexical steganography has existed for over a decade (Winstein 1999), there has been little research conducted in the area, with almost all existing solutions employing simple techniques based on synonym substitution (Alabish et al. 2013). A thorough search of the existing literature yielded only one paper that approached the problem from a machine learning perspective, Fang et al. (2017), who proposed a solution for the weaker problem of *cover generation* in which the user does not have control over the content of the cover text.

As a result, we will instead draw inspiration from the more mature field of Neural Machine Translation (NMT). Intuitively, lexical steganography can be thought of as a translation task, in that we are transforming the input text from one form to another while preserving the meaning. Instead of translating English text into the equivalent German text, we are translating English text into an equivalent English text that embeds the secret message.

Modelling the Problem

From this perspective, we can view lexical steganography as a problem of learning a generative model over the distribution of cover texts, such that given a particular cover text, we can generate new texts that are semantically equivalent. Embedding the hidden message can then be achieved through a learned algorithm that maps a particular codeword to a particular equivalent text. Decoding the hidden message from the stegotext is then simply a case of undoing this mapping.



	hash
~~i don't want my night to end.~~	0100
~~i just don't want my night to end.~~	0100
i just don't want my night to ending.	1100
~~i do not want my night to end.~~	0110
~~i really don't want my night to end...~~	0101
~~i just don't want my night to be over.~~	0001
i do not want my night to end...	1100
~~i really don't want my night to be over.~~	1101
~~i really do not want my night to end.~~	0110
i don't wanna my night to be over.	1100
~~i really don't want my night to stop.~~	0000
~~i just do not want my night to end...~~	0110
~~i really do not want my night to be over.~~	0101
i just do not want my night to be over.	1100
~~i just do not want my night to stop.~~	1110

Figure 1: An example of mapping semantically equivalent texts to codewords (Wilson et al. 2014). In this case, the codeword is determined by hashing the generated text.

We will now outline some examples of cutting-edge machine learning models and show how they can be modified for the purpose of lexical steganography.

Autoencoder

In its simplest form, an autoencoder is a feed-forward neural network consisting of an input layer, a single hidden layer, and an output layer of the same size as the input. The model is then trained to output a reconstruction of the original input, thereby learning a latent representation of the input within the hidden layer. Modified versions of this relatively simple model have been successfully used for a number of applications, such as variable-rate image compression (Toderici et al. 2017) and pretraining deep neural networks to have meaningful starting weights (Poultney et al. 2007). Compared to more traditional methods of unsupervised representation learning, such as principal component analysis, the autoencoder is shown to have a better expressive power, due to its ability to learn non-linear encodings (Goodfellow et al. 2016).

One type of autoencoder that is of particular use for lexical steganography is the variational autoencoder (VAE), which learns to encode a training sample as a probability distribution over the latent space, instead of a single point. This is effective in generalizing the autoencoder to act as a generative model over the training distribution (Kingma & Welling 2013). In the case of lexical steganography, a VAE could be trained on a set of cover texts such that at inference time, sampling from the distribution around a particular cover text will yield equivalent texts that can then each be mapped to a codeword.

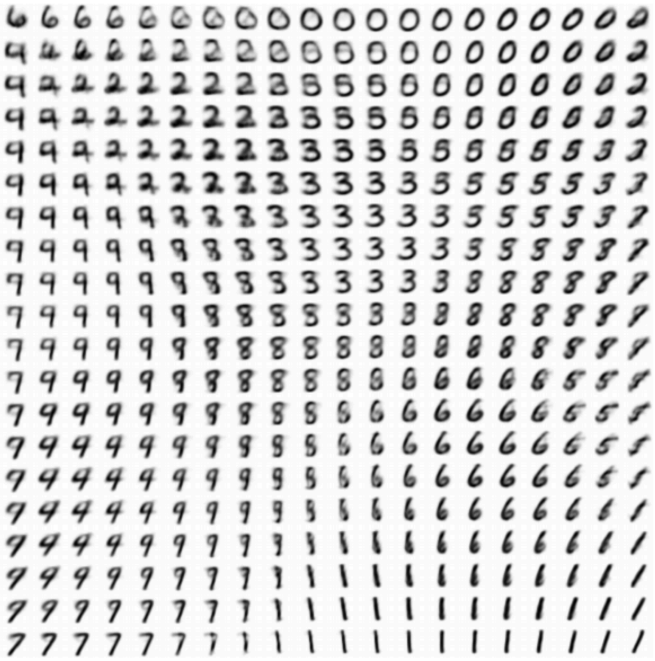


Figure 2: A visualisation of the manifold learned by a VAE trained on the MNIST handwritten digits dataset (Kingma & Welling 2013).

Seq2Seq

In order to be able to apply an autoencoder-like architecture to lexical steganography, we must first deal with the issue of varying input and output size. Unlike models that operate over images, in which it is reasonably acceptable to crop and resize the input to a specific dimension, our model should be capable of handling text of an arbitrary length. Furthermore, it should not necessarily be true that the size of our input text matches the size of the output stegotext. For example, inputting "Shall we go to the pub?" as a cover text could result in an output stegotext of "Do you want to go to the pub with me?".

Fortunately, recent research in NMT provides a solution in the form of the Seq2Seq model (Bahdanau et al. 2014). Similar to the autoencoder, it can be separated into an encoder RNN, which iterates over each token of the input sequence, modifying its internal state vector, and a decoder RNN, which takes the final hidden state of the encoder - the *context vector* and outputs an arbitrary number of tokens in the target language. A novel aspect of this model is the inclusion of a hidden state within the decoder which provides an *attention mechanism*, allowing the decoder to focus on different parts of the context vector as the output sequence progresses.

This model can be easily trained as an autoencoder, with Jang et al. 2018 providing a solution that modifies the Seq2Seq model to act as a variational autoencoder, showing improvements over a standard Seq2Seq autoencoder in the tasks of language modelling, missing word imputation, and paraphrase identification.

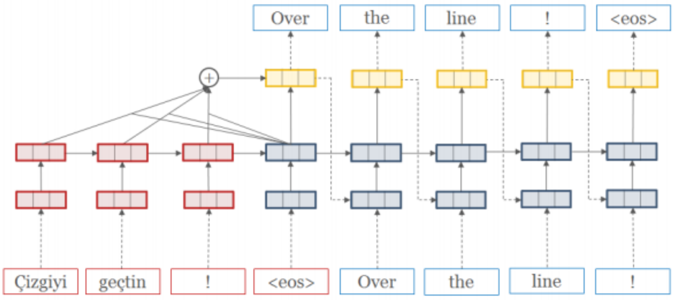


Figure 3: A visualisation of a Seq2Seq model for translating Turkish to English (Klein et al. 2017).

Generative Adversarial Network

An alternative model to the autoencoder is the Generative Adversarial Network (GAN), in which two networks, a generator and discriminator, are trained within a zero-sum game. The generator is given a random input and tasked with turning the input into a believable sample from the training set, while the discriminator is tasked with deciding if the given input is from the training set or the generator. As the discriminator learns to spot these 'fake' generator samples, the generator will learn to make better fakes, effectively learning the latent space of the training steganography.

For the similar task of image steganography, GANs have been used to embed a secret message within an image in a content-adaptive manner (Tang et al. 2017), although these systems do not yet give better performance than purpose-built algorithms such as S-UNIWARD (Holub et al. 2014). Regarding lexical steganography, the effectiveness of a GAN is limited by the discrete nature of the generator's output. This prevents the gradient from propagating smoothly from the discriminator to the generator.

Yu et al. 2017 propose a solution to this problem in the form of seqGAN , which models the generator as a stochastic policy in reinforcement learning. The reward can be determined from the discriminator output, which is then backpropagated through the generator using the REINFORCE algorithm and a Monte Carlo search to determine the reward for incomplete sequences.

Proposed Direction of the Project

Dataset

Our model will be trained on a corpus of English-language posts from Twitter. This choice of dataset comes with a number of advantages, most notably that many previous lexical steganography systems, such as CoverTweet (Wilson et al. 2014), are designed to work within the domain of Twitter, allowing for our system to be easily evaluated against existing methods. Furthermore, the casual context of Twitter allows for a greater steganographic capacity in the generated text. That is, misspellings and 'interesting' grammatical choices are more acceptable in a tweet than in a Wikipedia article or news report. This increases the number of equivalent texts that can be derived from a particular cover text and in turn allows for more information to be embedded.

Proposed Model

Our proposed model will consist of an encoder, a decoder and a discriminator for adversarial training. The encoder (Fig. 4) is based on a Seq2Seq autoencoder with the addition of a payload vector that is present at every stage of the encoding process. This model could also be extended to act as a VAE as per Jang et al. 2018. The goal of the encoder is to output a sequence that is similar to the input sequence, can be decoded to reveal the payload, and cannot be detected as containing an embedded payload.

The goal of similarity between cover and stegotext can be measured by a loss function that compares the two. This could simply be their cross-entropy, or a purpose-built measure such as their expected BLEU score (Auli & Gao 2014). To satisfy the remaining two goals, it is necessary to train a decoder (Fig. 5) that takes the stegotext and extracts the payload, and a discriminator (Fig. 6) that is given two sequences as input and must decide which sequence is the cover and which is the stegotext.

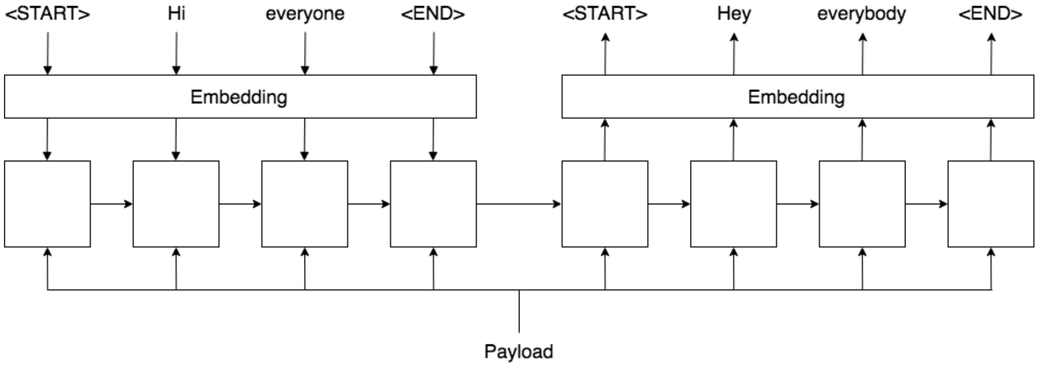


Figure 4: A visualisation of the proposed encoder

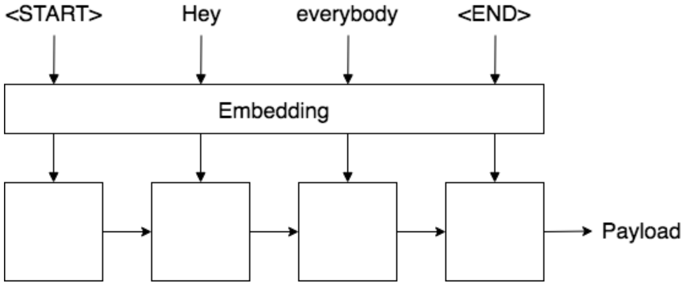


Figure 5: A visualisation of the proposed decoder

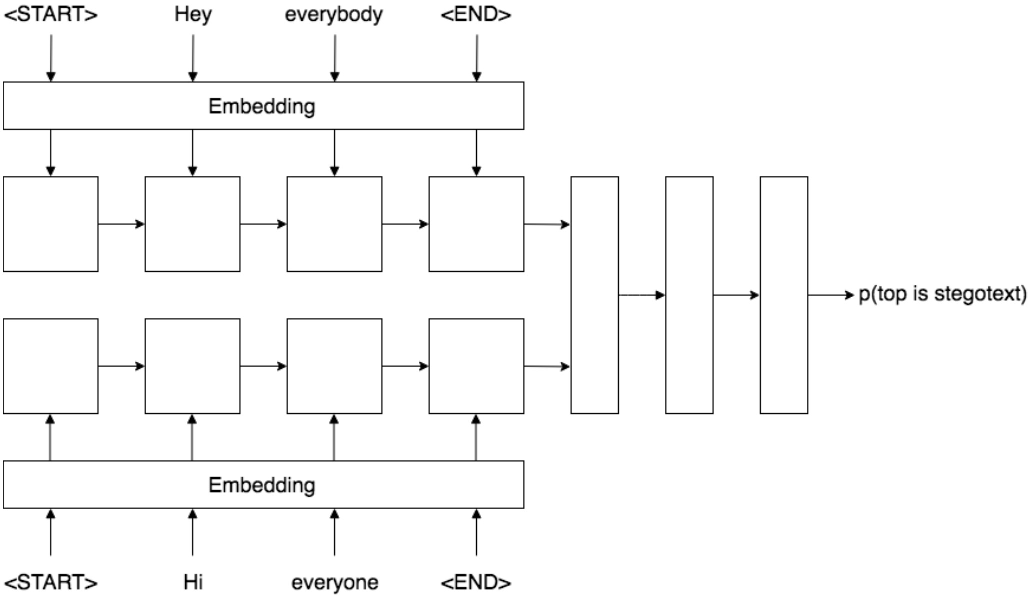


Figure 6: A visualisation of the proposed discriminator.

References

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

Auli, M. & Gao, J. (2014), Decoder integration and expected bleu training for recurrent neural network language models, *in* ‘Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)’, Vol. 2, pp. 136–142.

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

Fang, T., Jaggi, M. & Argyraki, K. (2017), ‘Generating steganographic text with lstms’, *arXiv preprint arXiv:1705.10742* .

Goodfellow, I., Bengio, Y., Courville, A. & Bengio, Y. (2016), *Deep learning*, Vol. 1, MIT press Cambridge.

Holub, V., Fridrich, J. & Denemark, T. (2014), ‘Universal distortion function for steganography in an arbitrary domain’, *EURASIP Journal on Information Security* **2014**(1), 1.

Jang, M., Seo, S. & Kang, P. (2018), ‘Recurrent neural network-based semantic variational autoencoder for sequence-to-sequence learning’, *arXiv preprint arXiv:1802.03238* .

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

Klein, G., Kim, Y., Deng, Y., Senellart, J. & Rush, A. M. (2017), ‘Opennmt: Open-source toolkit for neural machine translation’, *arXiv preprint arXiv:1701.02810* .

Poultney, C., Chopra, S., Cun, Y. L. et al. (2007), Efficient learning of sparse representations with an energy-based model, *in* ‘Advances in neural information processing systems’, pp. 1137–1144.

Tang, W., Tan, S., Li, B. & Huang, J. (2017), ‘Automatic steganographic distortion learning using a generative adversarial network’, *IEEE Signal Processing Letters* **24**(10), 1547–1551.

Toderici, G., Vincent, D., Johnston, N., Hwang, S. J., Minnen, D., Shor, J. & Covell, M. (2017), Full resolution image compression with recurrent neural networks., *in* ‘CVPR’, pp. 5435–5443.

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.

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

Yu, L., Zhang, W., Wang, J. & Yu, Y. (2017), Seqgan: Sequence generative adversarial nets with policy gradient., *in* ‘AAAI’, pp. 2852–2858.