Generative Models for Information Security:
A Machine Learning Approach to Lexical Steganography

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

Durham University

It was a beautiful sunny day

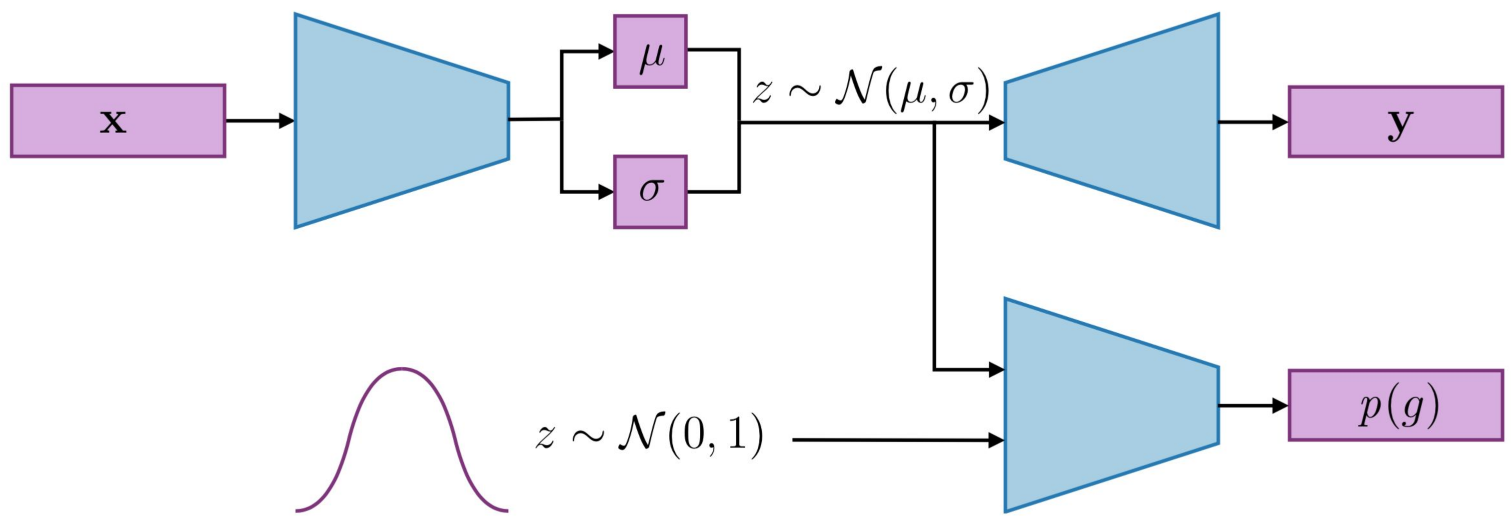
	hash
~~*It was a sunny day*~~	0101
It was a lovely morning	1101
~~*The sun was shining*~~	0001
~~*It was a wonderful sunny morning*~~	1001

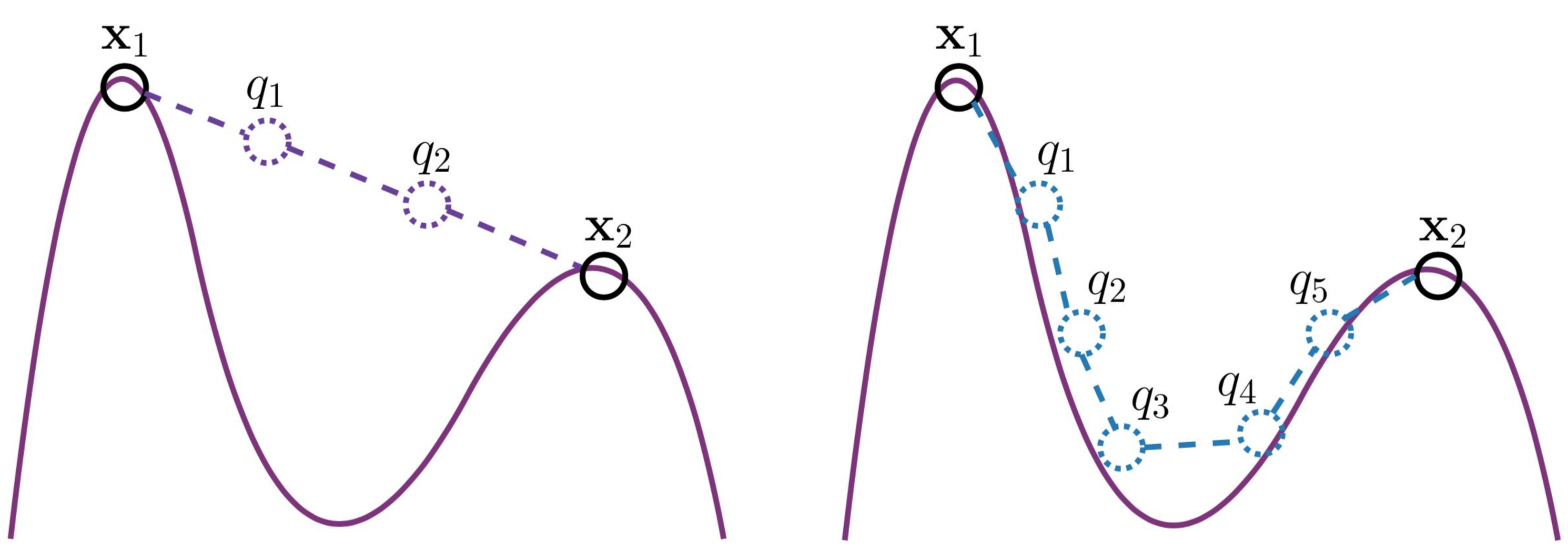
It was a lovely morning 1101

Lexical Steganography is the practice of hiding information within an inconspicuous **cover text** by exploiting the redundancies of natural language e.g. substituting words for their synonyms. This problem is closely linked to the fields of **machine translation** and **paraphrasing**.

Our system uses a **hash-based** approach, generating paraphrases of the cover text until its SHA-256 hash matches the intended payload. We investigate multiple approaches to generating these paraphrases, using a variety of models from the field of Natural Language Processing (NLP).

An **adversarial autoencoder** is used to encode the input sentence $\mathbf{x} = (x_1, \ldots, x_T)$ into the parameters to a **diagonal gaussian distribution**. From this distribution, we can sample a **latent vector** z, which can then decoded into a unique **stegotext** $\mathbf{y} = (y_1, \ldots, y_{T'})$.

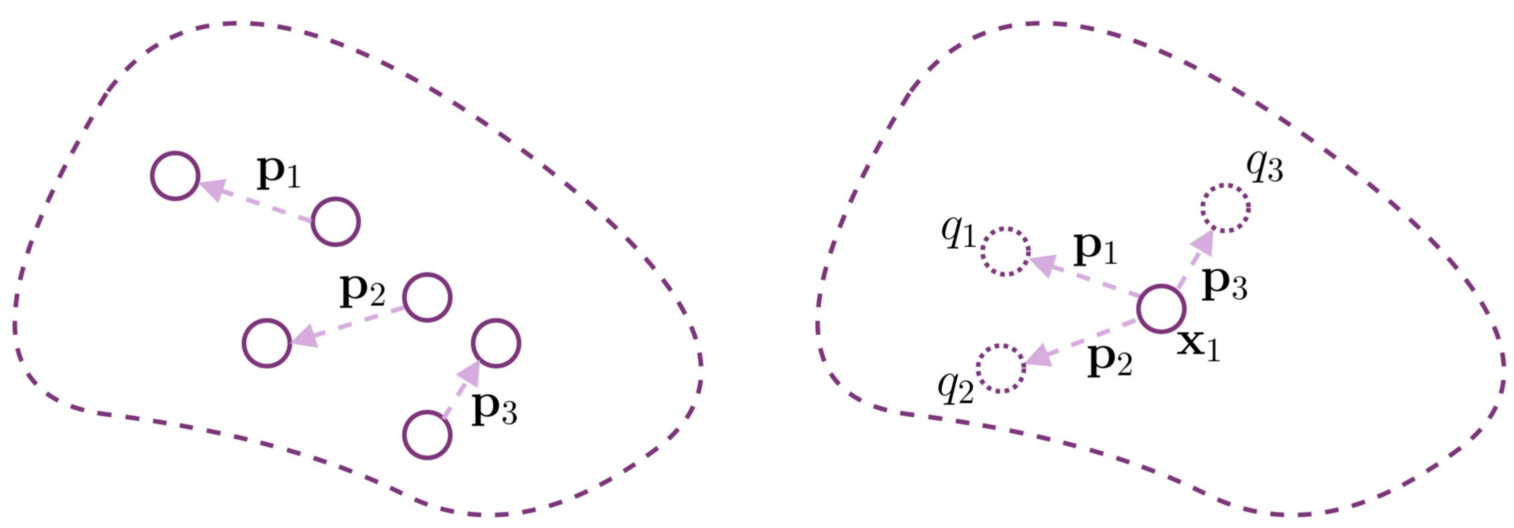


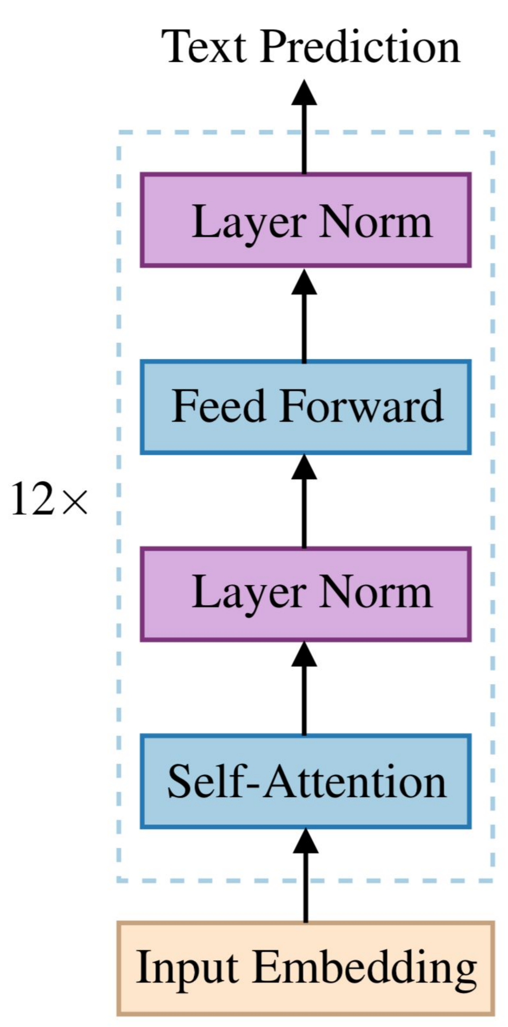


Gradient-based interpolation is used to generate stegotexts by travelling between latent codes while **remaining on the manifold**.

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

Analogical interpolation is used to generate stegotexts from known sets of paraphrase pairs. Taking their associated **paraphrase vectors** $\mathbf{p}$ in the latent space and applying them to $\mathbf{x}$.





$\mathbf{m}_{cond}$:

He's choking me.
he's strangling me.
—
Ben wore the palatinate gown.
Ben wore the purple robe.
—
Whoa.
stop right there.
—
Having my lunch!
at lunch!
—
Hello, Pearly.
hello, Pearly.
—
$\mathbf{x}$ *It was a lovely day.*

GPT-2 is a high-capacity general-purpose **language model**. A variety of tasks are encoded **within the input** $\mathbf{m}$, allowing it to learn **robust features** through **transfer learning.**

$$\mathbf{m} = (x_1, \ldots, x_T, \delta, y_1, \ldots, y_{T'})$$

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

We use a specially crafted input $\mathbf{m}_{cond}$ to **encode the paraphrase generation task**, encouraging GPT-2 to output paraphrases of $\mathbf{x}$.

Automatic and human evaluations show that the GPT-2 conditioning approach achieves a **state-of-the-art capacity,** successfully embedding a 4 bit message into 91.4% of samples while remaining **indistinguishable from fluent English** in over 75% of samples