Linguistic Steganography on Twitter: Hierarchical Language Modelling with Manual Interaction

Alex Wilson, Phil Blunsom, and Andrew D. Ker

Oxford University Department of Computer Science, Parks Road, Oxford OX1 3QD, UK.

ABSTRACT

This work proposes a natural language stegosystem for Twitter, modifying tweets as they are written to hide 4 bits of payload per tweet, which is a greater payload than previous systems have achieved. The system, CoverTweet, includes novel components, as well as some already developed in the literature. We believe that the task of transforming covers during embedding is equivalent to unilingual machine translation (*paraphrasing*), and we use this equivalence to define a distortion measure based on statistical machine translation methods. The system incorporates this measure of distortion to rank possible tweet paraphrases, using a hierarchical language model; we use human interaction as a second distortion measure to pick the best. The hierarchical language model is designed to model the specific language of the covers, which in this setting is the language of the Twitter user who is embedding. This is a change from previous work, where general-purpose language models have been used. We evaluate our system by testing the output against human judges, and show that humans are unable to distinguish stego tweets from cover tweets any better than random guessing.

1. INTRODUCTION

This paper is concerned with the problem of hiding data (steganographic content usually called *payload*) inside a cover consisting of English language. More specifically, our system is tailored towards hiding in *tweets*, short social media messages of up to 140 characters; these are typically heavily abbreviated, and have features such as *usernames* and *hashtags*.

Hiding in language is dissimilar to hiding in digital media such as pictures and video. In the latter, adding small amounts of noise – incrementing pixel values or quantized transform-domain coefficients, for example – is typically invisible in most cases. As long as one takes care not to add high-frequency noise into low-frequency parts of a cover, visual imperceptibility is easy and contemporary digital media steganography can focus on evading *statistical* detection. The opposite is true of language: even slight changes to a sentence often produce clearly incorrect results (grammatically, semantically, or violating consistency), and the literature has had to use powerful models of language while making very limited changes to the text. There is also an unsolved problem of coding and synchronization. Statistical steganalysis of language is underdeveloped but has not proven very powerful; it is currently more difficult to evade a human Warden than a statistical one.

We propose a natural language stegosystem for tweets, modifying them as they are written to hide 4 bits of payload per tweet. This work includes some novel components as well as some already developed in the literature. In particular, the system incorporates two measures of distortion: an automatically-computed hierachical language model to rank possible tweet paraphrases, and human interaction to pick the best. Experiments, testing the output against human judges, show that humans are unable to distinguish stego tweets from cover tweets any better than random guessing.

The paper is structured as follows. In subsection 1.1 we briefly describe the *Twitter* social media platform. In section 2 we survey existing methods for natural language steganography and outline the main components of our stegosystem. In section 3 we describe how the components are combined and applied to the particular domain, including some examples of language transformation and their results in tweets. In section 4 we describe the evaluation of our system by human Wardens, and analyse their ability to discriminate cover and stego tweets. Finally, we give directions for further research.

Further author information:

A. D. Ker: E-mail: adk@cs.ox.ac.uk, Telephone: +44 1865 283530

A. Wilson: E-mail: awilson@cs.ox.ac.uk

1.1 Twitter

Twitter is a social networking platform, launched in 2006. Users of the service post short messages of up to 140 characters (*tweets*); this type of messaging is known as *microblogging*. By default, these tweets are publicly visible: it is not even necessary to have a Twitter account to read them. Each user has a username, and tweets can be directed at another user by placing '@' in front of their username somewhere in the tweet. A unique feature to Twitter is the ability to *re-tweet* a message; re-posting another user's message to spread the message, echo the sentiment, or add a comment. Such tweets are usually marked as such with 'RT' placed at the start of the tweet.

Tweets can be categorised using a special word called a *hashtag*; these are words prepended by the '#' (hash) symbol, usually – but not exclusively – placed at the end of a tweet. It is possible to search Twitter according to these hashtags. A commonly used example is '#fail', placed at the end of a tweet indicating the message is discussing a failure of some sort.

Twitter is currently one of the ten most visited websites in the world.[1] The company announced in October 2012 that 500M tweets are sent each day, from 200M regular users.[2] A study on usage of social networking sites[3] found that 18% of internet users in America claimed to use Twitter in some way.

The average number of tweets per user is 307. Until their account was suspended in late 2013, the user with the greatest number of tweets was 'Yougakudan_00' (36,307,144 tweets as of May 2013): the majority were apparently-random binary strings, which might have been part of an interesting system to transmit data (but not a form of steganography).

We are interested in using Twitter as a channel for covert communication, so we analysed a collection of tweets provided by the Harvard TweetMap[4] to estimate certain statistics of tweet contents: average number of words used per tweet, range of vocabulary per user, etc. The corpus consists of tweets posted throughout the month of May in 2013, and contains 72M tweets from 1.8M users in America, Canada, the United Kingdom, Australia and New Zealand (chosen to limit the tweets primarily to English language). Figure 1 shows our findings.

Ideally we should like to know the entropy of a tweet (since, if perfect embedding were possible, the payload size would be bounded above by it[5]), but we are not aware of any previous work providing such an estimate. From our corpus we estimate that the *trigram* entropy of tweets is approximately 7.45 bits per word. We estimate that the average number of words per tweet is 12.9, and the average size of a user's vocabulary is 8.6 words per tweet.

2. NATURAL LANGUAGE STEGANOGRAPHY

This section introduces the main aspects of, and problems facing, natural language steganography; it also gives a description of language modelling. We propose a distortion measure for natural language stego, using aspects of statistical machine translation.

2.1 Transformations

Steganography in digital media works by identifying components which are irrelevant to the semantics of the content, typically low-level noise. Similarly, natural language (NL) steganography aims to hide information in natural language by applying a transformation to a cover text that does not alter its meaning*. This transformation is required to preserve the meaning and grammatical correctness of the original sentence. Previous natural language stegosystems have used synonym substitution,[7] paraphrase substitution,[8] and sentence structure manipulation.[9,10]

A number of these transformation techniques are available thanks to research into natural language watermarking; any transformation that is appropriate for watermarking is almost certainly appropriate for steganography. Given this close relationship between the two areas of research, there is a notable tendency for steganographic schemes to be mislabelled as watermarking (e.g. Ref. 11), or for the aims of watermarking (namely robustness)

*Although it might be possible to perform pure *cover generation* steganography, this is likely to be as difficult for natural language as any other medium, except in particular domains,[6] so the literature focuses on *cover modification*.

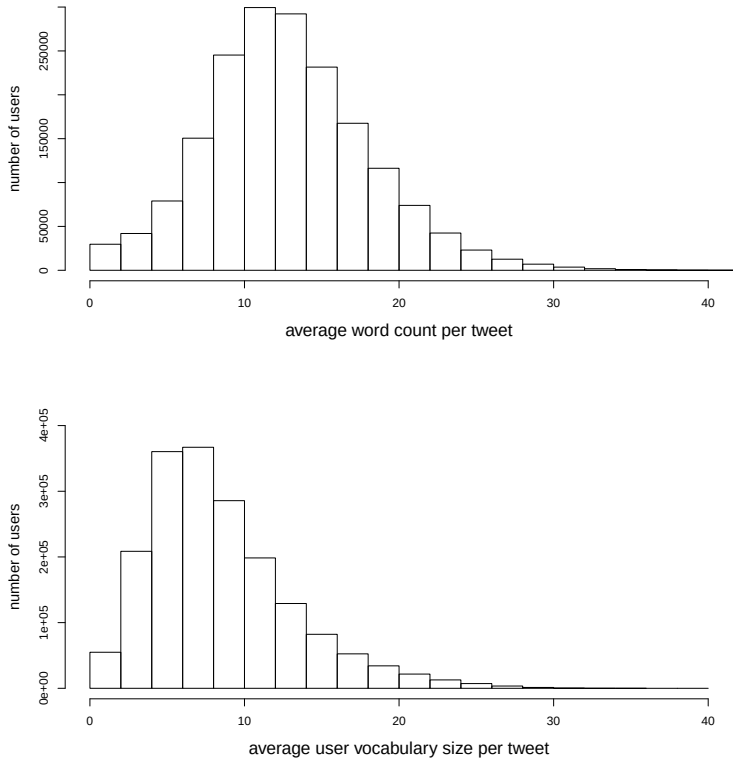

Figure 1. A selection of tweet statistics extracted from the Twitter corpus.

to leak into steganography (e.g. Ref. 12). The goals of the two topics are quite distinct, and beyond shared transformations they should not be confused.

NL stegosystems use these transformation techniques to generate a number of possible stego objects from each cover object; regardless of the technique used, the stegosystem must then assign a value to each of these possible stego objects, in order to carry the payload. The receiver must be able to determine the value carried by the stego object, and this is difficult because the cover (or the set of possible transformations of it) is typically not recoverable from the stego object. Unlike steganography in digital media, for language there is no simple analogue of the least-significant bit or remainder modulo n: the NL steganography literature has struggled with this problem, which is essentially one of side information (the cover) not available to the decoder.

Even more difficult is that capacity is typically not constant. Assuming that text has been broken down into units such as sentences (and that the embedding process does not desynchronize this decomposition!), depending on the embedding procedure it is typical for some units to have more transformation options – and therefore be able to carry more payload – than others. Some units may be unable to carry any payload at all, if no suitable transformations can be found. The latter presents a problem of synchronization, often approached by restricting the possible substitutions to guarantee that the reverse transformation produces the same set of possibilities as the original.[7,13] Another attempt used a graph colouring scheme to assign values.[8] Both sacrifice a great deal of potential capacity.

Here, we will simply carry payload by a keyed hash of the tweet (subsection 3.3). This does not require the receiver to perform reverse transformation, but it does not solve the synchronization problem. Finding a proper solution to this was beyond the scope of this work, but is of utmost importance to NL steganography. We have also sacrificed some capacity, since different stego tweets will not all have distinct hashes. This is famously

related to the so-called *coupon collector's problem*: in our case of hiding 4 bits per tweet, assuming uniform hash output, the *mean* number of distinct stego tweets needed to cover all 16 possible payloads is approximately 55.

We view any transformation of cover text as equivalent to the task of unilingual machine translation (also referred to as *paraphrasing*). Machine translation (MT) aims to translate text in a source language to text with equivalent meaning in a target language; unilingual MT is the case where the source and target languages are the same. In NL steganography, the source text is the cover, and we want to translate it into our target stego text. The observation of this fact allows us to borrow aspects of MT for use in NL steganography; in the next section we suggest a possible distortion measure based on statistical machine translation.

2.2 A Natural Language Distortion Measure

The first stegosystems, in both digital media and natural language, treated all changes as equally applicable: they would embed any change that conveyed the correct payload. Subsequent systems introduceded contextual checks to filter out the most inappropriate changes (be they visible high-frequency noise in images, or unacceptable natural language transformations), but treated all other changes as being equally valid.

Modern so-called *adaptive* steganography refined this idea further, using a measure of *distortion*: each possible change is associated with a distortion cost, and a coding method is used to minimize the average distortion. To date, adaptive steganography only exists for digital images, and developing distortion for NL steganography is a desirable prerequisite to developing mature coding methods for this domain.

We build a distortion function for NL steganography by borrowing the fundamental probability from statistical machine translation: $\Pr(e|f)$, the probability that a target sentence e is a translation of a source sentence f. By regarding the stego text (s) as the target sentence, and the cover text as the source sentence (c), we write this as $\Pr(s|c)$.

We can use the negative log likelihood of translating c to s as our distortion function, which gives us a suitable value of infinity when the probability of translation is 0 (and therefore that the stego text is not a translation of the cover text). We also need the distortion to be 0 when no change is made: we get this by adding the log likelihood of $\Pr(c|c)$, i.e. the log probability of the cover text translating to itself. Note that this is only suitable if $\Pr(c|c) > \Pr(s|c)$ for all possible values of s, but this should normally be the case. Thus our distortion measure is:

$$d(c, s) = -\log \frac{\Pr(s|c)}{\Pr(c|c)}$$

When using this to rank possible stego objects, we don't have to divide by $\Pr(c|c)$, because it is constant.

In order to estimate this distortion, we must model $\Pr(s \mid c)$. This is difficult to do directly,[14] so instead we apply Bayes' law,

$$\Pr(s|c) = \frac{\Pr(c|s)\Pr(s)}{\Pr(c)}$$

The denominator is constant for any given cover, so we can ignore it:

$$\Pr(s|c) \propto \Pr(c|s)\Pr(s)$$

The probability of $\Pr(c|s)$ is known as the *translation model probability*. These probabilities are dependent on the transformation scheme, and no attempt is made here to discuss methods of estimation. Ideally, the translation model would be trained dependent on the cover source: the probability of a translation is not likely to be the same for two arbitrary cover sources. For example: the probability of translating 'I am' to 'I'm' should almost certainly be higher when the covers are casual e-mails rather than academic papers, where contraction is rarely used. For the moment we use a fixed translation model, and investigation of personalized translation models is left for future work.

The second factor, $\Pr(s)$, is the *language model probability*. Here we are modelling the language of our stego objects, which (we hope) should be practically identical to that of the cover source. The use of a language model in a stegosystem is not a new idea, but using it explicitly as part of a distortion function is. In previous work

the language model has been a general English model (trained on a large non-specific dataset); for any system with a specific cover source, this is not good enough. The next section gives an overview of how we can build language models.

2.3 Statistical Language Modelling

We want to model the probability of a sentence s, which is a sequence of words $w_1, \ldots, w_T$ ($\Pr(w_i, \ldots, w_T)$). We can decompose this probability using the chain rule, giving:

$$\Pr(w_i, \ldots, w_T) = \Pr(w_1) \prod_{i=2}^{T} \Pr(w_i | w_1, \ldots, w_{i-1})$$

It is not practical to know all the conditional probabilities, so we have to limit the *history* (the conditioning context $w_1, \ldots, w_{i-1}$) to only a few words. For example, when the history is limited to a single word, we approximate:

$$\Pr(w_1, \ldots, w_T) \approx \Pr(w_1) \prod_{i=2}^{T} \Pr(w_i | w_{i-1})$$

This is called an n-gram language model, where $n-1$ is the number of words used for the history; these are $(n-1)^{\text{th}}$ order Markov models. A general formula for these models is:

$$\Pr(w_1, \ldots, w_T) \approx \Pr(w_1) \prod_{i=2}^{n} \Pr(w_i | w_{i-1}, \ldots, w_1) \prod_{i=n}^{T} \Pr(w_i | w_{i-1}, \ldots, w_{i-n}) \tag{1}$$

In order to approximate the probability of each n-gram ($\Pr(w_i \mid w_{i-1}, \ldots, w_{i-(n+1)})$), we can use a maximum likelihood estimation (MLE):

$$\Pr(w_i | w_{i-1}, \ldots, w_{i-n+1}) = \frac{\text{count}(w_{i-n+1}, \ldots, w_i)}{\text{count}(w_{i-n+1}, \ldots, w_{i-1})}$$

where $\text{count}(w_i, \ldots, w_{i-n+1})$ is the number of times the sequence of words $(w_i, \ldots, w_{i-n+1})$ has been seen in some training text. For example, for a bigram ($n = 2$) model,

$$\Pr(w_2 | w_1) = \frac{\text{count}(w_1, w_2)}{\text{count}(w_1)}$$

In order to evaluate how well a language model performs on some test data, we use a measure called perplexity. This is a measure of how well the language model predicts the data, and is calculated as $2^{H(s)}$, where $H(s) = -\frac{1}{n} \log \Pr(s)$ is the cross-entropy, s is our test data and n is the number of words in s.

2.4 Smoothing

The problem with using the MLE for approximating the probability of an n-gram occurs when the n-gram is unseen in the training data: it will be assigned a probability of 0. Language models are complex, with lots of probabilities to estimate (one for every possible n-gram), which means that we need a vast amount of data to adequately train the model. In practice, we are very unlikely to have enough, and there will always be unseen n-grams; it is almost certainly not appropriate to say these are all impossible. This is a case of the model over-fitting to the training data and we use *smoothing* techniques to adjust the counts of n-grams in order to better approximate the probabilities of unseen n-grams.

A number of smoothing techniques exist.[15] The simplest approach is *add-one* or *additive* smoothing, where we add 1 (or some other constant δ to every count). For a bigram model, adding one gives:

$$\Pr(w_2 | w_1) = \frac{1 + \text{count}(w_1, w_2)}{|V| + \text{count}(w_1)}$$

where $|V|$ is the size of some given vocab V (a set of possible words).

In practice this is not a good method.[16] A more sophisticated method is that of Good-Turing smoothing.[17] Although this is not the current best, it is the basis upon which some newer methods are built. For an n-gram seen r times, we adjust the count to r^*:

$$r^* = (r+1)\frac{n_r + 1}{n_r}$$

where n_r is the number of n-grams that are seen r times in the training data. The probabilities are then estimated with:

$$\Pr(w_i|w_{i-1}, \ldots, w_{i-n+1}) = \frac{r^*}{\sum_{r=0}^{\infty} n_r r^*}$$

for an n-gram $w_{i-n+1}, \ldots, w_i$ that has been seen r times.

When we have a very small language model (e.g. for a domain where gathering in-domain data is expensive or difficult), this type of smoothing will be unable to improve the model sufficiently. Instead, it is possible to combine language models hierarchically to improve performance; in the small model example, it could be combined with a larger general model. Section 3.2 describes one such construction.

2.5 Human Warden and Manual Interaction

It is a unique aspect of NL steganography that, for the majority of stegosystems, the most powerful attacker is human.[18] This is due to limitations of natural language processing (NLP) algorithms, which have not yet reached the point where they can compete with humans' in-built sense of language. Humans are particularly good at spotting the type of problems that NL stegosystems are most prone to making: mistakes with word sense (the meaning of a word) and sentence fluency. For applications without security concerns, these are somewhat acceptable; for steganography, they are not. Their presence would signal to any human reader (Kerckhoff's attacker) that the text has been changed by some automatic system. (A human attacker is indeed plausible in this domain, see subsect. 3.)

There are three ways for a NL stegosystem to avoid making these mistakes. We could solve the problem of word sense disambiguation (knowing the sense of a word in a given context), but this would require a significant advance in natural language modelling. We could minimise detectable mistakes by making only very restricted changes (e.g. Ref. 19), but this sacrifices most of the covers' capacity. Or we can ask the human steganographer to intervene during embedding, in order to guide the system toward generating stego text without any word sense or fluency issues. We take this last approach, which we could express as a second measure of distortion: the first is given by the language model, and used to rank potential stego objects for the second, the 'human-in-the-loop' who selects the most fluent.

A number of stegosystems have been developed using this method,[12,20–22] but most are based around cover-generation steganography, where the human is allowed to change generated covers so that the payload is preserved but the object is fluent. These systems have little practical use, because the generated covers make little sense in context: it is unlikely therefore that they can be used by any steganographer without giving away the presence of a hidden message.

The literature mainly takes the view that any stego text generated with a 'human-in-the-loop' is automatically undetectable, though there is little evaluation to confirm this. It is our opinion that human interaction does *not* guarantee this security. Humans can ensure that word sense and fluency issues are avoided, but leave open the possibilities of statistical (e.g. word frequency) attack. Evaluation of these types of stegosystem is scant: it is not even known whether using a 'human-in-the-loop' during embedding makes steganography impervious to a *human* attacker, something we will investigate in this paper.

3. COVERTWEET

Twitter is an excellent setting for exploring natural language steganography: the covers all come from one specific source (the steganographer's Twitter account); stego objects are nicely self contained messages, of a fixed maximum length; and tweets are posted at the time of writing, so messages have to be embedded at the time of writing and it is reasonable to make use of the steganographer during embedding.

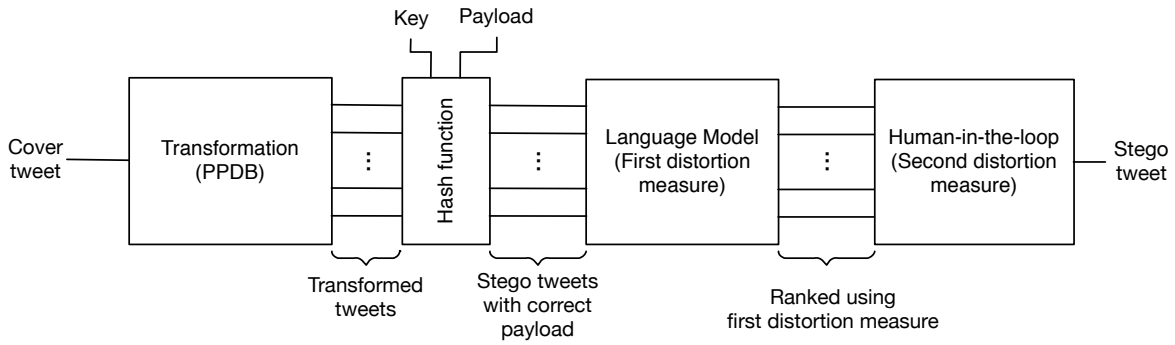

Figure 2. The CoverTweet embedding process.

Twitter has been used for steganography before,[23] but this was done by combining a number of existing, non-natural language techniques (e.g. hiding data in images attached to tweets and in message whitespace). Steganography on microblogging sites might be desirable to a number of users. In China, the site Sina Weibo is roughly equivalent to Twitter. There are a number of forbidden words that get messages deleted, and users banned; users have taken to using code words as a way to discuss banned topics.[24] Each message is checked by a human; the Beijing Times reported that 2M censors are employed by the government.[25] When attempting steganography, we can consider these censors as being human Wardens, and 2M wardens would be enough to read every single tweet posted on a given day.

Our formulation of NL Twitter steganography is as follows: the steganographer, Alice, is a Twitter user with some number of previously posted innocent tweets. Alice wishes to start hiding information in her tweets. Bob, the receiver, knows Alice's username, and has access to her tweets. By knowing that the cover source is one specific user, our language model we use to create our distortion measure must specifically model *Alice's* Twitter language.

Our proposed natural language stegosystem for Twitter is called *CoverTweet*, and it works as follows:

1. Alice generates a cover tweet.
2. The system generates a number of possible stego tweets from the cover.
3. The system selects only stego tweets that convey the desired payload (have the correct hash).
4. The selected stego tweets are ranked using a distortion measure.
5. Alice chooses the best stego tweet from this shortlist.

Bob decodes the payload by applying the same keyed hash function. Figure 2 illustrates the embedding process. Each part of the system is described in greater detail in the following subsections.

3.1 The Translation Model

The system transforms the cover object of these using the recently released Paraphrase Database (PPDB).[26] This is a database of paraphrase rules, extracted from a number of parallel bilingual corpora using the approach detailed in Ref. 27; this uses the assumption that if two english strings translate to the same foreign string, they share the same meaning. The same technique was used to create a paraphrase dictionary in Ref. 8, though on a much smaller scale.

Each paraphrase rule contains a source string, a target string, and some features of the rule (including the translation probability of the rule). The exact format of each line is:

```
LHS ||| SOURCE ||| TARGET ||| (FEATURE=VALUE )* ||| ALIGNMENT
```

where LHS is the constituent label for the source and target (e.g. noun phrase, verb phrase), and ALIGNMENT describes how the words in SOURCE map to words in TARGET (neither of these were required here).

The database comes in multiple sizes, the largest containing a total of 169M rules. For the stegosystem, we only used a fraction of this total amount, using the middle sized database, and restricting the rules to the lexical (one word) and phrasal (multiword) paraphrase rules (i.e. not including the syntactic paraphrases that contain nonterminal symbols – parts-of-speech tags, rather than words); this resulted in a total of 3.4M rules.

The stegosystem transforms cover tweets by searching the database for any rule with a source string that matches a word or phrase in our cover; the system constructs the possible stego tweets by applying every possible combination of paraphrase rules. For example, the tweet (taken from our Twitter corpus):

```
my grandma trevino just asked when she was going to see my beautiful gf again.  uhh bad news grams
```

might result in the following stego tweets:

```
my grandmother trevino just asked when she was going to see my beautiful gf again.  uhh bad news grams
my grandma trevino just asked when she was going to see my beautiful gf again.  uhh really bad news grams
my grandmother trevino just asked when she was going to see my beautiful gf again.  uhh really bad news grams
```

The top stego example shows the result of applying the rule that 'Grandma' can be substituted for 'Grandmother', and the subsequent examples both apply the rule that 'bad news' can be substituted for 'really bad news'. The former rule appears in the PPDB (with some features removed) as:

```
[NN] ||| grandma ||| grandmother ||| ... p(f|e)=1.24811,p(e|f)=1.42164 ... ||| 0-0
```

It is common for Twitter users to address tweets to multiple recipients by placing multiple usernames at the start; likewise to categorise tweets with multiple hashtags at the end. The paraphrase rules were augmented with twitter specific rules taking advantage of this: any blocks of hashtags or usernames can be reordered without changing the meaning of the tweet. This is certainly true of usernames, and is often appropriate for hashtags as well (for example, the common hashtags '# fail' and '# lol' are equally valid used together in either order).

We rank the possible stego objects using our distortion function:

$$d(s, c) = -\log\left(\frac{\Pr(c|s)\Pr(s)}{\Pr(c|s)}\right)$$

where $\Pr(c|s)$ is our probability of translating a stego tweet s to a cover tweet c. The probability $\Pr(c|s)$ is approximated by assuming each substitution in a given tweet is independent, allowing us to calculate from the PPDB:

$$\Pr(c|s) \approx \prod_{i=1}^{T} \Pr(c_i|s_i)$$

where T is the number of phrases in the tweet, and s_i is the substitution for the word or phrase c_i in the original tweet.

3.2 The Language Model

We have already made the assumption that Alice has a number of innocent tweets that we can use to train a language model, and in this work we use the trigram model ($n = 3$ in (1)). For training we used the same corpus that we used to evaluate user statistics in subsection 1.1. The problem is that Alice alone almost certainly does not have enough cover data for adequate training, whereas we can get a vast amount of general Twitter data; this is a problem of domain adaptation.

Our approach is to create a hierarchical language model: a combination of models, where one is given priority but others help with unseen n-grams. We train a small model on Alice's tweets (holding back a small tuning set);

and a general one on a large amount of Twitter data from other users. Sophisticated approaches to this problem exist[28,29] but we take a simple approach of combining the two models by linearly interpolating the probabilities from both models. For example, the probability of a bigram becomes:

$$\Pr(w_2|w_1) = (1-\lambda)\Pr_A(w_2|w_1) + \lambda \Pr_G(w_2|w_1)$$

where $\Pr_A(w_2|w_1)$ is the bigram probability from a model trained on Alice's data, and $\Pr_G(w_2|w_1)$ is the probability from a general model. When the system comes across a word that Alice hasn't used before, the probability can be estimated from the general data. We can tune the system by finding the value of λ that minimises the perplexity of the interpolated language model on the held back tuning set of Alice's tweets.

3.3 Assigning Values

To ensure that the sender and receive unambiguously agree on the assignment of stego tweets to payload values, we borrow the method used by translation-based stegosystems.[30–32] We use a keyed hash function, which removes even the need for the receiver to know the substitution rules available to Alice: only the shared key is required.

Given the large number of possible stego objects for a given cover, it is computationally expensive to generate every single possibility in order to filter down to the desired options by hashing. Luckily, we do not require the hash to be cryptographic, which allows for efficient searching. For a given tweet, the value is determined by getting a keyed hash digest for each individual word (here using 4 bits from a hash generated using the MD5 algorithm), bitwise rotating each value according to its position in the tweet, then exclusive-or-ing these values together. Partial hash-values are calculated for each index of the tweet. A table is constructed, mapping an index and a possible hash value to paraphrase substitutions that will result in the given hash value up to this index in the tweet; these substitutions are paired with the partial hash value required up to the previous index in the tweet. Using this table, it is possible to construct only the stego sentences that provide a desired hash value in linear time.

Though simple, assigning values using a hash function is not without its own problems. The payload size must be a fixed number of bits, regardless of the number of options available for a particular cover. As well as wasting potential capacity, it may fail to embed certain payloads in certain covers that have only a few appropriate translations.

3.4 Human Interaction: A Secondary Distortion Measure

There may usually be multiple possible stego tweets that convey a desired payload; we rank these using our distortion measure. If the quality of the measure were sufficient we could select the stego tweet with the lowest distortion, but limitations of NPL suggest that this will not capture all aspects of detectability. Specifically, the lowest-distortion option is not guaranteed to be fluent or without word sense mistakes. The final selection of the stego tweet is instead made by Alice: they are shown a ranked list of options, and choose which stego object to send.

We view this 'human-in-the-loop' as a secondary distortion measure, attenuating the shortcomings of the first. But the use of a human exacerbates the problem of not being able to use every cover, if they rule out all possible substitutions. When the translation engine finds substitutions with the correct hash, it is often because the original tweet is too short, or contains phrases that do not have associated rules in the PPDB; the receiver could conceivably create a rule to ignore any tweets with these properties. This would not be possible when the failure to embed is only because the human ruled out all substitutions.

This is another synchronization problem and beyond the scope of this work: for now, we simply assume that Bob can determine which tweets were used for steganography, or that Alice re-words any cover tweet that would have failed to embed. But essentially the problem is one of side information not available at the decoder, which should be amenable to coding methods.

```
Original:  If I never hear Ignition by RKelly again it wouldn't bother me
        1  if i never hear powerup by rkelly again it would n't bother me              -44.531
        2  well , if i ever hear ignition by rkelly again it would n't mind me         -46.163
        3  maybe if i ever hear contact by rkelly again it would n't bother me         -46.611
        4  well , if i never hear ignition by rkelly again it would n't matter to me   -46.657
        5  if i ever hear ignition by rkelly again it would not bother me              -46.850
        6  well , if i never hear lit by rkelly again it would n't mind me             -47.146
        7  look , if i never hear ignition by rkelly again it would n't mind me        -47.150
        8  maybe if i ever hear powerup by rkelly again it would n't mind me           -47.180
        9  well , if i never hear power-up by rkelly again it would n't bother me      -47.343
       10  though i 'd never hear ignition by rkelly again it would n't bother me      -47.387
       11  well , if i ever hear ignition per rkelly again it would n't bother me      -47.555
       12  maybe if i ever hear powerup per rkelly again it would n't bother me        -47.609
       13  maybe if i never hear powerup by rkelly again it would n't matter to me     -47.674
       14  if i never listen ignition by rkelly again it would n't bother me           -47.794
       15  if i never hear lit by rkelly again it would not bother me                  -47.833
       16  unless i never hear ignition by rkelly again it would n't disturb me        -48.390
       17  well , if i ever hear lit by rkelly again it would n't matter to me         -48.392
       18  well , if i never hear lit per rkelly again it would n't bother me          -48.396
       19  look , if i ever hear ignition by rkelly again it would n't matter to me    -48.396
       20  ...  if i ever hear contact by rkelly again it would n't bother me          -48.418
```

Figure 3. Some example output of the system. The 20 possible stego tweets with the lowest distortion; the log probability of each stego tweet is included. In total the system generated 71407 options for this cover tweet.

3.5 Parsing

We processed the tweets from the Harvard TweetMap[4] before using them for training the language models, and applied similar processing to the cover tweets input by Alice.

Each tweet was canonicalised to lowercase, because the PPDB does not distinguish cases in any of the paraphrase rules. Our hash function was designed to be case independent, so if required Alice could reintroduce capitals when choosing the final stego tweet. In principle the system could attempt to reintroduce capitals to the stego tweet, so that it mirrors the cover, but this was not attempted at this stage.

Tweets were also tokenised (using a tokeniser provided by the Natural Language Toolkit (NLTK)[33]), which separates punctuation from words; such tokenisation is required to use the PPDB, and for training the language model. This step also separates contracted forms of words (e.g. separating '`shouldn't`' into the two tokens '`should`' and '`n't`'). For transmission, we reversed this by removing the spaces that were added during the tokenisation. As for cases, the hash function was designed to be space independent, so Alice could add or remove spaces if necessary. For the special case of hashtags, we separated the hash symbol from the categorising word, so that the language model could learn which words are likely to appear before a hashtag, and which words are likely to occur as hashtags.

For training the language model, all usernames and URLs were replaced by general tokens ('`USER`' and '`URL`'). Effort was made to remove retweets, by ignoring all tweets containing the token 'RT'; this was necessary to ensure that the language model was not training twice on the same content.

3.6 Sample Output

Figure 3 shows some of the output presented to Alice by the prototype CoverTweet system, during embedding. Each tokenised stego tweet (for a specific payload) is displayed, along with its log translation probability according to the hierarchical language model. This example highlights the need for the secondary distortion measure: the best substitution is probably the one ranked 4th, as the first three are inappropriate for various reasons: fluency issues and words used in incorrect senses.

Figure 4 contains some examples of cover and stego tweets, with the changes highlighted.

```
people still wash there air force 1s shoes, i take it some people didnt learn the lesson....  if you
                          wash air force 1s they will turn yellow!!  lol
people still wash there air force 1s shoes, i guess some people didnt learn the lesson.  if you wash
                            air force 1s they will turn yellow!!  lol
```

Figure 4. An example of a steganographic tweet with the unmodified cover

4. EXPERIMENTAL RESULTS

We began by selecting ten users from the corpus, on which to perform experimental evaluation. We selected users at random meeting the following criteria:

1. They had posted (during the month for which the corpus was collected) between 500 and 1000 tweets. This was sufficient to train a language model and few enough to filter spam accounts.
2. Their average number of words per tweet was at least 9. This removed users who frequently posted messages too short to contain payload.
3. The size of their vocabulary per tweet was approximately equal to 8.5. This also removed spam accounts and collected 'typical' users.

Removing the data for these users from the corpus, we trained the general language model on the rest (which consisted of approximately 75M tweets). For each user, we split their tweets up into three sets: the cover set, containing the most recent 100 tweets; the set for training the language model, containing 90% of the other tweets; and a tuning set, containing the remaining 10%. We used the SRI Language Modelling Toolkit (SRILM)[34] to train each model, and find the optimal interpolation weights: the weight for interpolating each individual's language model with the general model was chosen so that it minimised the perplexity of the language model on the tuning set. Stego tweets were then generated from the tweets in the cover set.

4.1 Evaluation of Language Model

Our first task is to check that the hierarchical language model is working properly. We calculated the perplexity of the combined language model on the cover set for each user, while changing the interpolation weight, and show the mean of the results in Figure 5. The interpolated models had better fit (lower perplexity on the tuning set) than the baseline model, though the average interpolation weight that gave the lowest perplexity was higher than expected. We attribute this to the number of tweets we had for each user; the user models are very small, and had an extremely high perplexity on the test set (the average perplexity was8803.45). We would expect the optimal weight to be lower for users with significantly more tweets (which might be the case if the tweets in our corpus were not collected over the course of a single month). Further work is required in this area: there is scope to apply more sophisticated approaches to domain adaptation. When the distortion measure is defined using the language model probabilities, the estimated probabilities need to be as accurate as possible, to ensure that an attacker does not have a better measure of distortion.

4.2 Evaluation of Stegosystem Security

One of the main goals of our system was to be secure against human Wardens, while still embedding a reasonable amount of data. In order to evaluate how successfully the system achieved this goal, we designed an experiment that used human judges to attempt to spot stego tweets.

We used the CoverTweet system to generate 20 stego tweets for each of the 10 users. On average it took 50 tweets from the set of covers to find 20 usable tweets, with each tweet containing 4 bits of randomly generated payload. Note that this does not necessarily mean each cover is only capable of embedding an average of 1.6 bits; it only means that 60% of the covers are not capable of embedding 4 bits, and we do not know exactly how many bits could be embedded. Of the 20 cover tweets that were changed successfully, it is possible some of these would actually have been capable of embedding more than 4 bits.

For the evaluation, we gathered twenty human judges, who were each given an exercise: having been shown 40 known 'innocent' tweets (taken from the training and tuning sets), they were presented with 20 unclassified

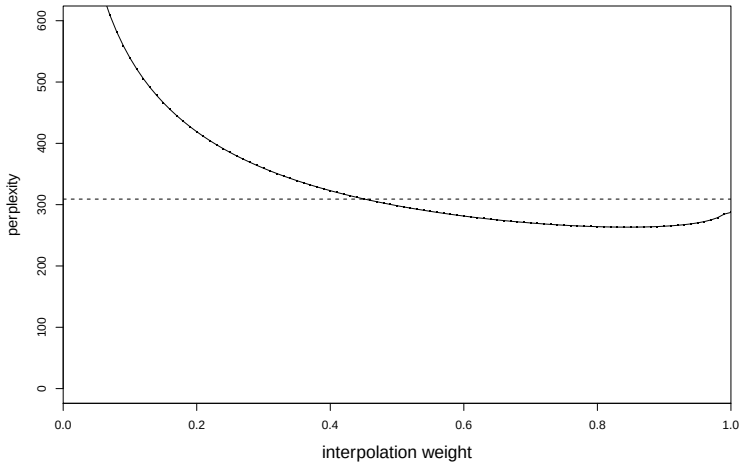

Figure 5. Averaged perplexity of the language model on the testing data for each actor, as the interpolation weight changes. The dotted line shows the average perplexity for the general model. At weight 0, the general model is unused, and the average perplexity is 8803.45.

tweets from a particular user. Of these unclassified tweets, 10 are stego (picked at random from the 20 stego tweets generated), and 10 are innocent: the task was for each judge to identify the 10 stego tweets. They were asked to do this by assigning a score (1-10, to denote how sure they were about a particular tweet being stego) to 10 of the 20 tweets. By ensuring that all the tweets were capable of embedding the 4 bit payload, judges were not able to simply choose the longer tweets, or tweets containing particular features that allow for easy changes (such as multiple usernames).

After assigning 10 scores, the judges were given another 40 innocent tweets, and were asked whether this affected their scores at all: this was to test whether having more history would affect the judge's ability to detect steganography. Each judge repeated this task for 5 of the users; the users were divided up so that each was seen by the same number of judges. We then measured the ability of the judges to distinguish cover and stego tweets, and whether this ability differed significantly between judges and between twitter users. Figure 6 shows part of a task presented to a judge.

The judges were recruited from students and staff at Oxford University. Not all were computer scientists, and none were computational linguistics experts. They were aware of the general steganographic aim of the CoverTweet system, but not its exact mode of operation. This mimics a real-world scenario where a human censor monitors Twitter traffic without targetting a specific stegosystem.

4.3 Evaluation Results

It turned out that the judges' performance was in no (statistically significant) way affected by seeing only 40 or the full 80 innocent tweet history of the users they were scoring. Therefore we include only the results for the latter, which is the more stringent testing environment for the security of our stegosystem.

The accuracy of each identification of the stego tweets, for each user individually, is displayed in Fig. 7 (left). The same results broken down by judge are in Fig. 7 (right). Overall, out of 2000 classifications, 1037 were correct and 963 were incorrect.

Overall accuracy of human judges

These 2000 answers are not entirely independent because the judges were told to identify exactly 10 guilty tweets out of 20 from each actor: we can account for this dependence by measuring their accuracy only on true guilty tweets (of necessity their true negative rate equals their true positive rate). Our judges collectively identified 515

Innocent tweets:

1. your avi @user41 >>>>
2. middle finger up to my competition
3. im gone download me some emojis #thestruggle lml
4. @user67 the one you wore lastnight to the fashion show thing? lml.
5. i got the jokes tonight
6. phone sticky <<<.
7. she on time to get off.
8. tssk, the struggle is real she was not gone talk to her until she seen me tweeting her smh.
9. but anywaysssss ..
10. get of my tl please ya.
11. fake? haha lets not go there hun.
12. you change alot i dont even no who you is anymore.
13. don't start something you can't finish.
14. i don't argue over petty stuff ill just start to distance myself away from you ..
15. """@user84: fake girls <<<<<<<<<<<<<<<<<<<<<<<<<<<< no can do"""
16. i need to call daija asap.
17. modeling classes ova the summer >>>
18. leave me alone ..... lol.
19. lml.
20. i no she telling my other cousin a whole different story but i could careless.
21. my circle geting smaller everyday, this thing is geting worse. im cuting em' off.
22. im going to get off that case and im gone dismiss it, anit nobody got time for that. it is what it is.
23. you must care lol .. hold up? why am i even talking to you or responding to you?
24. from what i heard today i dont even wanna talk to her no more!
25. i shall return .. bbl.

10 steganographic and 10 innocent tweets:

1. soon as i got home im gone eat than go sleep i had a long day!!
2. i need to buy me some new headphones..
3. would beilive me if i said im inlove? baby ..
4. my mama bout me another new pair of beats but... it's not the headphones it the ear ones..
5. people still wash there air force 1s shoes, i take it some people didnt learn the lesson.... if you wash air force 1s they will turn yellow!! lol
6. yall must have 11:11 set 1 minute early before yall tweet it, because soon as 11:11 hit yall don't wastes no time. lol
7. i sleep with my earings on but tonight i can't .
8. #oomf toes is ugly .... woooo jesus everytime i see them things i run, i sweaaa!
9. @user270 lol.
10. @user341 i'm remembering that time in 7th grade when ashley b made us get that f on that project in science class, lmao we been to mad!
11. most of the people who got mouth can't beat you .
12. #weallknowthatonepersonwho hate on you, don't like you, envy you, laugh in ya face and then talk about you!!! i know em'
13. why my family choose to come my house this late at night? i can't deal with all that laughing right now .. lol
14. feel like going with my cousin today .. i need a laugh .
15. """@user163: that awesome moment when you say something really funny and everyone laughs, so you just sit there like a boss."""
16. you can tell when somebody hating on you!
17. mfs be thinking i gaf about whatever they think, when really i don't .
18. people say ice not good thing for you, yet little do they no ice is just frozen water. people need to think before they speak.

Figure 6. An example of the tweets shown to the judges. The steganographic tweets have been highlighted.

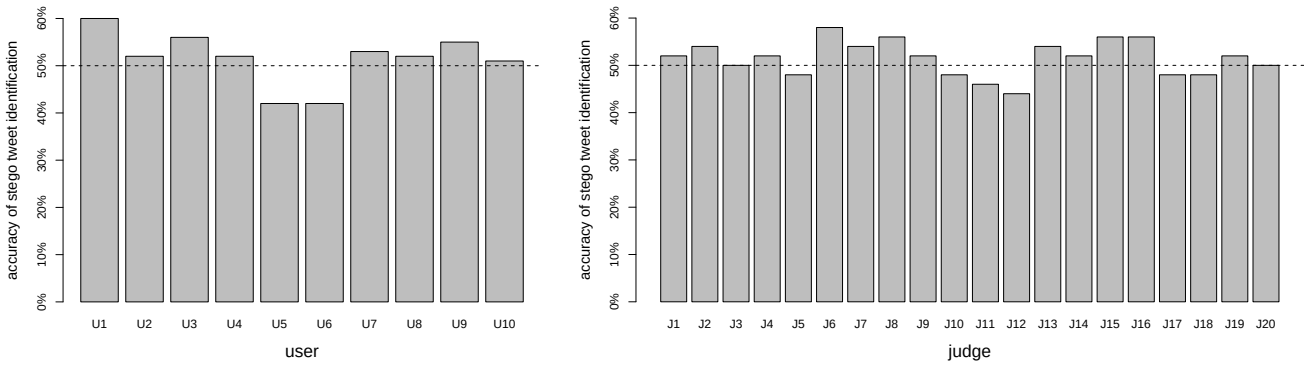

Figure 7. Accuracy of detection of stego tweets by human judges. Each judge scored a total of 100 tweets; each user was scored on 200 tweets. Left, accuracy for each user; right, accuracy of each judge.

out of 1000 guilty tweets correctly. A simple Z-test allows to test the null hypothesis that judges are guessing randomly, verses a one-sided alternative that they are answering correctly more often. There is insufficient evidence to reject the null hypothesis of random guessing ($p = 0.180$).

When we designed the system, we were not expecting it to be so secure! Part of the security stems from the frequency of spelling and grammar mistakes in the cover tweets: Twitter seems a good candidate for steganography because its users already inject a lot of randomness into it.

Effects on accuracy

We wish to know whether some judges were (statistically significantly) more accurate than others, whether some users' tweets were more or less difficult to classify, and also whether identification with higher confidence scores tended to be more accurate. All of these can be accomplished in one go using an analysis of variance, the results of which are displayed in the following table:

Factor	d.f.	deviance	p-value
Judge	19	14.1707	0.7737
User	8	19.6039	0.0112*
Confidence	1	0.5431	0.4612

We conclude that neither the identity of the judge, nor the confidence with which they make judgements, significantly effects the accuracy of classification. We also tested separately whether higher confidence was associated with correct answers, using a Wilcoxon rank sum test. This is makes no assumptions about the distribution of the scores, but does use their ranking. Again no association was found. ($p = 0.621$).

To investigate further the effect of user, we performed a multivariate logistic regression, modelling the log odds of a correct judgement in terms of user and judge. We omit the regression table because only one factor was significantly different from 0: user 5, it turns out, was significantly more difficult to classify correctly than the rest (but no other actors differed significantly). We attribute this to the particular selection of unclassified tweets for this actor, though potentially exacerbated by a noticeable lack of consistency in the user's language.

Two of actor 5's innocent tweets had features that a number of judges took as clear signs of guilt, but were just a product of this unpredictable nature: one tweet was part of a group of three tweets to the same two users, but the usernames of the recipients were in reverse order; the other tweet used both 'you' and 'u' in the same sentence (rather than just using one form). In addition, two of the stego tweets had the sort of changes that were difficult to spot by humans (e.g. substituting '..' with '...').

5. CONCLUSION

We have presented a method for hiding information in tweets. It involves two types of distortion in a novel combination: a language model, which can be computed automatically, and human involvement to select the 'optimal' stego tweet from a list of possibile paraphrases of the cover. We found the NL transformations via unilingual translation, applying a large (multi-word) paraphrase database. This combination is more sophisticated than previously found in the NL steganography literature, allowing us to hide 4 bits in usable tweets, where usable tweets represented approximately 40% of our corpus: a very small amount of information, but comparable to other NL steganography methods which typically manage one or two bits per sentence.[19,35] It is likely that the capacity of our system can be increased in future work, by the addition of more domain-specific paraphrase rules and also by finding coding methods that can adapt better to the different capacities of covers.

We should perhaps note that, if the estimate of 500M tweets per day[2] is accurate, and even if every tweet can *and is* used to hide steganographic payload of, say, 4 bits, the *total global covert bandwidth of Twitter* is still only an average of 2.83KB/s. This is not a high-capacity covert channel, but it has the potential to be a very secure one.

In future work we hope to attack the coding and synchronization problems. Whether an individual tweet is usable at all is an example of side information at the encoder, and it is an application of "wet paper codes"[36] (where the symbols are the bits embedded in each tweet) to avoid changing unusable tweets without that information being required at the decoder. The main challenge is to avoid using the human oracle too often during the embedding process. However we could do much better, in terms of overall capacity per tweet, if we were able to embed a different number of symbols in each tweet: more in tweets with more potential paraphrases. This will require a novel coding approach.

The NL steganography literature has often failed to measure the security (undetectability) of proposed systems, particularly when steganography is mis-labeled as NL watermarking. In this work we recruited human judges to test the detectability of CoverTweet output, with a string of negative results: they did not detect CoverTweet significantly better than random guessing, and there is nothing to indicate that any factor (except for one particularly difficult actor) changes this. This is excellent news for the security of our stegosystem. It suggests that there is indeed room for higher capacity, but also that we should perhaps re-visit our assertion that human judges are the most accurate detectors of NL steganography. However, the literature on statistical steganalysis of NL stegosystems is extremely sparse, with most detectors targeted at very old steganographic methods,[37,38] so there is nothing practical to test against. In principle the steganalysis literature could implement a similar hierarchical language model to subsection 3.2 and use average perplexity as a simple detection metric. The availability of training data from matched sources is, most likely, even more essential here than in image steganalysis.

Finally, we note the difficulty of performing large-scale experiments when human judges are involved. It is quite simple to run millions of steganalysis tests on a computer, but the two thousand scores gathered for this study relied on considerable goodwill by a number of volunteers. One option is to crowdsource the judges, via a marketplace such as Amazon's *Mechanical Turk*, something recently used by researchers in computational linguistics.[39,40]

ACKNOWLEDGMENTS

We thank the human judges for volunteering to complete their steganalysis tasks.

REFERENCES

[1] http://www.alexa.com/topsites, Alexa Internet (2013).

[2] Smith, C., "By The Numbers: 68 Amazing Twitter Stats," http://expandedramblings.com/index.php/march-2013-by-the-numbers-a-few-amazing-twitter-stats, Digital Marketing Ramblings (2013).

[3] Duggan, M. and Smith, A., "Social Media Update 2013," http://pewinternet.org/Reports/2013/Social-Media-Update, Pew Internet & American Life Project (2013).

[4] Mostak, T., "Harvard TweetMap," http://worldmap.harvard.edu/tweetmap, Harvard (2013).

[5] Wang, Y. and Moulin, P., "Perfectly secure steganography: Capacity, error exponents, and code constructions," *IEEE Transactions on Information Theory* **55**(6), 2706–2722 (2008).

[6] McKellar, D., "Spammimic," http://www.spammimic.com/ (2000).

[7] Winstein, K., "Lexical steganography through adaptive modulation of the word choice hash," http://web.mit.edu/keithw/tlex (1998).

[8] Chang, C.-Y. and Clark, S., "Linguistic steganography using automatically generated paraphrases," in [*Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*], 591–599, Association for Computational Linguistics (2010).

[9] Chang, C.-Y. and Clark, S., "The secret's in the word order: Text-to-text generation for linguistic steganography," in [*Proceedings of COLING 2012*], 511–528, The COLING 2012 Organizing Committee (December 2012).

[10] Topkara, M., Riccardi, G., Hakkani-Tür, D., and Atallah, M. J., "Natural language watermarking: Challenges in building a practical system," in [*Proceedings of SPIE 6072, Security, Steganography, and Watermarking of Multimedia Contents VIII*], 60720A–60720A, International Society for Optics and Photonics (2006).

[11] Halvani, O., Steinebach, M., Wolf, P., and Zimmermann, R., "Natural language watermarking for german texts," in [*Proceedings of the first ACM workshop on Information hiding and multimedia security*], 193–202, ACM (2013).

[12] Wyseur, B., Wouters, K., Preneel, B., and Leuven, K., "Lexical natural language steganography system with human interaction," in [*Proceedings of the 6th European Conference on Information Warfare and Security*], 303–312 (2007).

[13] Bolshakov, I. A., "A method of linguistic steganography based on collocationally-verified synonymy," in [*Information Hiding*], 180–191, Springer (2005).

[14] Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L., "The mathematics of statistical machine translation: Parameter estimation," *Computational linguistics* **19**(2), 263–311 (1993).

[15] Chen, S. F. and Goodman, J., "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language* **13**(4), 359–393 (1999).

[16] Gale, W. and Church, K., "What is wrong with adding one," *Corpus-based research into language* **1**, 189–198 (1994).

[17] Good, I. J., "The population frequencies of species and the estimation of population parameters," *Biometrika* **40**(3-4), 237–264 (1953).

[18] Grosvald, M. and Orgun, C. O., "Free from the cover text: a human-generated natural language approach to text-based steganography," *Journal of Information Hiding and Multimedia Signal Processing* **2**(2), 133–141 (2011).

[19] Chang, C.-Y. and Clark, S., "Adjective deletion for linguistic steganography and secret sharing.," in [*Proceedings of COLING 2012*], 493–510, The COLING 2012 Organizing Committee (2012).

[20] Munoz, A., Gallardo, J. C., and Alvarez, I. A., "Improving n-gram linguistic steganography based on templates," in [*Proceedings of the 2010 International Conference on Security and Cryptography (SECRYPT)*], 1–4, IEEE (2010).

[21] Grosvald, M. and Orgun, C. O., "Free from the cover text: a human-generated natural language approach to text-based steganography," *Journal of Information Hiding and Multimedia Signal Processing* **2**(2), 133–141 (2011).

[22] Muñoz, A., Carracedo, J., and Alvarez, I. A., "Hiding short secret messages based on linguistic steganography and manual annotation," in [*2010 IEEE 10th International Conference on Computer and Information Technology (CIT)*], 960–964, IEEE (2010).

[23] Zirikovic, Z., http://revolucionlibrary.wordpress.com, Secretwit (2013).

[24] "How do sina weibo users avoid censorship?," http://www.digitalintheround.com/sina-weibo-censorship, Digital in the Round (2013).

[25] "China employs two million microblog monitors state media say," http://www.bbc.co.uk/news/world-asia-china-24396957, BBC News (2013).

[26] Ganitkevitch, J., Van Durme, B., and Callison-Burch, C., "PPDB: The paraphrase database," in [*Proceedings of NAACL-HLT*], 758–764, Association for Computational Linguistics (2013).

[27] Bannard, C. and Callison-Burch, C., "Paraphrasing with bilingual parallel corpora," in [*Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*], 597–604, Association for Computational Linguistics (2005).

[28] Daumé III, H. and Marcu, D., "Domain adaptation for statistical classifiers.," *Journal of Artificial Intelligence Research* **26**, 101–126 (2006).

[29] Wood, F. and Teh, Y. W., "A hierarchical nonparametric bayesian approach to statistical language model domain adaptation," in [*International Conference on Artificial Intelligence and Statistics*], 607–614 (2009).

[30] Grothoff, C., Grothoff, K., Alkhutova, L., Stutsman, R., and Atallah, M., "Translation-based steganography," in [*Information Hiding*], 219–233, Springer (2005).

[31] Stutsman, R., Atallah, M., Grothoff, C., and Grothoff, K., "Lost in just the translation," in [*Proceedings of the 21st Annual ACM Symposium on Applied Computing (SAC 2006)*], (2006).

[32] Meng, P., Hang, L., Chen, Z., Hu, Y., and Yang, W., "STBS: A statistical algorithm for steganalysis of translation-based steganography," in [*Information Hiding*], 208–220, Springer (2010).

[33] Bird, S., Klein, E., and Loper, E., [*Natural language processing with Python*], O'reilly (2009).

[34] Stolcke, A. et al., "SRILM-an extensible language modeling toolkit.," in [*INTERSPEECH*], (2002).

[35] Chang, C.-Y. and Clark, S., "Practical linguistic steganography using contextual synonym substitution and vertex colour coding," in [*Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*], 1194–1203, Association for Computational Linguistics (2010).

[36] Fridrich, J., Goljan, M., and Soukal, D., "Wet paper codes with improved embedding efficiency," *IEEE Transactions on Information Security and Forensics* **1**, 102–110 (2006).

[37] Taskiran, C. M., Topkara, U., Topkara, M., and Delp, E. J., "Attacks on lexical natural language steganography systems," in [*Electronic Imaging 2006*], 607209–607209, International Society for Optics and Photonics (2006).

[38] Chen, Z.-l., Huang, L.-s., Yu, Z.-s., Zhao, X.-x., and Zhao, X.-l., "Effective linguistic steganography detection," in [*Computer and Information Technology Workshops, 2008. CIT Workshops 2008. IEEE 8th International Conference on*], 224–229, IEEE (2008).

[39] Callison-Burch, C. and Dredze, M., "Creating speech and language data with amazon's mechanical turk," in [*Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*], 1–12, Association for Computational Linguistics (2010).

[40] Callison-Burch, C., "Fast, cheap, and creative: evaluating translation quality using amazon's mechanical turk," in [*Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*], **1**, 286–295, Association for Computational Linguistics (2009).