Bring Linguistics Back to Cryptanalysis: Using Attestation to Break the Advanced Encryption Standard

Madeline Boese
Computational Linguistics Lab
University of Florida
mboese@ufl.edu

Abstract

The Advanced Encryption Standard (AES) is currently considered the preferred standard of encryption for secure messaging by the National Institute of Standards and Technology. While its predecessor, the Data Encryption Standard (DES), can be analyzed in mere seconds with modern cryptanalysis algorithms, those same algorithms are unfeasible for cryptanalysis of AES. This is primarily because the key, or the list of values used during encryption, was increased from a length of 56 bits to 128 bits. The list of all possible combinations, also called the probability space, is tens of magnitudes larger in AES than DES. Current DES cryptanalysis methods use mathematical methods to find the most statistically likely approximate key in the entire probability space. The increased probability space in AES is too large to find an approximation using these formulas. However, these methods operate under the assumption that every key is possible. While this is mathematically true, when looked at from the lens of a linguist, many of these keys create messages that are impossible in language.

Linguistic attestation is the documentation that a word, grapheme, sound, or other linguistic feature exists in a language. This thesis proposal presents an algorithm that eliminates AES keys using linguistic attestation. I apply the idea that any grammatical message encrypted by AES will not contain any "unattested" data in its input. The algorithm proposes trimming the probability space by removing keys that are unattested. Once a smaller probability space has been created, any method that searches the probability space for a solution may be applied to find the key from the new, smaller list of filtered keys.

1 Introduction

Cryptology is a two-sided science focused on creating ways to secure information between a source and an intended recipient using encryption (cryptography) and discovering ways for non-recipients to decrypt and access that information (cryptanalysis) (NIST, 2019; Coron, 2006). Cryptography and cryptanalysis are at constant odds with each other; as cryptography develops new ways to secure information, cryptanalysis requires the development of new exploitations to extract that information, which in turn pushes cryptography to develop even stronger encryption schema (Hannan and Asif, 2017).

Historically, cryptologic algorithms, called ciphers, were computed by hand and were fairly simple (Ycart, 2012). These ciphers were analyzed using statistical linguistic knowledge of the message's language, such as analyzing the average distribution of graphemes in a language, and calculating the "Index of Coincidence" (the likelihood the same grapheme is randomly chosen twice in a string of text) (Jara Vera and Sanchez Avila, 2019; Ycart, 2012).

The advent of computers in WWII revolutionized the science of cryptology from a grapheme-based science to a mathematical and computational science (Hannan and Asif, 2017). Modern cipher algorithms, including the Advanced Encryption Standard, often encrypt information by mapping it to a matrix of bytes, known as a "blockcipher" (Coron, 2006). This information is then manipulated with binary level operations that are optimal for computer calculation, like exclusive-or (XOR) and modulo (Hannan and Asif, 2017; NIST, 2023). Regardless of the shift to binary manipulation, the fundamentals of ciphers remains the same: information is formulaically transposed or substituted according to a key (the list of values that replace variables in a cipher) (Jara Vera and Sanchez Avila, 2019). Therefore, it stands to reason that the *linguistic-based analysis* that formed the basis of cryptanalysis should still be a functional tool in modern systems; yet modern cryptanalysis largely ignores linguistic properties in favor of statistical or mathematical methods (NIST, 2019).

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop), pages 375–382

My thesis proposes that *reintroducing* statistical linguistic analysis to the field of cryptology can make progress towards the cryptanalysis of the Advanced Encryption Standard (AES). Specifically, I introduce the use of attestation to filter all possible keys of an encrypted message into a set of all linguistically possible keys. Attestation is the documentation that a feature, sound, word, or other property exists in a language. Current algorithms search through all keys on the basis that they are all mathematically possible, *but completely ignore linguistic possibility*.

AES's largest strength over its predecessor is its immense key size (Devi and Kotha, 2019). The lowest possible number of potential keys for any given message in AES is 2^{128} or approximately 3.4×10^{38}. This can be represented in what is known as a "probability space," which in this case contains 128 values, each with 2 options of equal likelihood (0 or 1). The benefit of applying a linguistic perspective to computer cryptography is the ability to trim the probability space with low-cost tests of keys. One of the current fastest method to decrypt the predecessor to AES, the Data Encryption Standard (DES), leverages neural networks to optimize a likelihood attack called "linear cryptanalysis" that approximates the most statistically likely keys and identifies the true key from the results (Sharmila and Karuppasamy, 2023). The main issue employing linear cryptanalysis to AES is that the number of statistically likely keys increases as the number of possible keys increases. The probability space of DES (2^{56}) is not even $\frac{1}{10^{21}}$ of AES's probability space. However, if enough keys could be eliminated from AES, the probability space could become small enough that linear approximation methods become possible again.

Linguistics analysis can be used for probability space trimming because any grammatical message will not contain any unattested data. Therefore, any key that would imply an original message contains unattested data can be eliminated from the probability space. This principle forms the foundation of my proposed filtering algorithm. In what follows, Section 2 further explores why linguistics is a necessary tool for cryptanalysis from a historical perspective, and connects this claim to current limitations of modern cryptanalysis work; Section 3 reviews the AES algorithm; in Section 4 I introduce my proposed algorithm; Section 5 and 6 present conclusions and limitations, respectively.

2 Related Work

2.1 Evolution of Cryptanalysis

Some of the first recorded instances of the applied study of cryptanalysis dates back to ancient Arab scholars (Ycart, 2012). Scholar Al Kindī proposed that given the alphabet of a language, a large text can be gathered and the frequency of its graphemes can be extracted and compared to the frequency of the graphemes in an encrypted text (Ycart, 2012). This technique, known as frequency analysis, is a foundational technique throughout the history of cryptanalysis (Hannan and Asif, 2017; Ycart, 2012).

As an evolution in cryptanalysis causes ciphers to become less secure, a need for stronger ciphers arises (Hannan and Asif, 2017; Devi and Kotha, 2019). Frequency analysis works well for monoalphabetic ciphers where each grapheme is substituted with one other grapheme, such as the famous "Cesar cipher" where each letter in the alphabet is shifted by a given number to a different letter (Hannan and Asif, 2017). In response, poly-alphabetic ciphers were developed where each grapheme is substituted with a different variable grapheme that is dependent on some other variable, like position in the message (Hannan and Asif, 2017). Thus, the "Index of Coincidence" was developed, which is a linguistic measure of distribution that calculates the likelihood that two randomly chosen graphemes from a text will be the same grapheme (Ycart, 2012; Hannan and Asif, 2017).

Despite the increase in encryption strength, when ciphers needed to be decrypted, linguistic based methods continued to be developed or modified to meet the needs of cryptanalysts (Hannan and Asif, 2017; Ycart, 2012). While this is partially due to those sending encrypted messages preferring to use basic ciphers until the mid 20th century (Hannan and Asif, 2017), cryptanalysis as a science was largely based on identifying patterns preserved between a language and its form when encrypted (Jara Vera and Sanchez Avila, 2019). It only changed to a purely mathematical science recently after the advent of using computer automation for encryption during World War II (Hannan and Asif, 2017).

2.2 Modern Standards for Effective Encryption

The introduction of computers and the progression into an age of digital information exchange

changed the methodologies of cryptography (Hannan and Asif, 2017). The information that needed to be encrypted became digital, and ciphers transitioned to computer algorithms. The Data Encryption Standard, or DES was the standard in cryptology until the year 2001 (Devi and Kotha, 2019). It is arguably the predecessor to AES, but it is no longer considered secure due to its small key size (Devi and Kotha, 2019). Over the last few years, many papers have come out that exploit its relatively small number of possible keys and newer technology to consistently break DES ciphers (Sharmila and Karuppasamy, 2023). Notably, the methods employed to decrypt DES are all based on sorting through the entire probability space using various mathematical and statistical principles, or machine learning (Sharmila and Karuppasamy, 2023).

DES cryptanalysis has advanced far enough that the National Institute of Standards and Technology (NIST) started seeking a new cipher that would resist the cryptanalysis of DES at the end of the 20th century (Devi and Kotha, 2019). This led to the creation of AES, which was announced as the new standard of secure encryption in 2000 and published in 2001 (NIST, 2023; Devi and Kotha, 2019). DES had a key that was 56 bits in length, while AES can have a key length of 128 bits, 192 bits, or 256 bits (Devi and Kotha, 2019; NIST, 2023). The increase in key length alone increases the probability space from DES to AES by over 10^{21} times, even at AES's smallest size.

2.3 Current Methods for Modern Cryptanalysis

As previously mentioned, there has been an increase of literature focused on utilizing advancements in technology for DES cryptanalysis in the past few years. The following works illustrate different methods in the literature across both DES and AES. Despite the amazing efficiency these cryptanalysis algorithms achieve decrypting DES, or the promising success they show towards partial AES decryption, AES's large probability space continues to outpace advancements in the technology.

Hermelin et al. (2019): Hermelin et al.'s method is an extension of Matsui's statistical linear attack (Matsui, 1994), which exploits a linear relation between plain-text, cipher-text and the key. Matsui's method employs two algorithms that work in a fraction of cases, which is generally referred to as "linear approximation". Hermelin et al. evaluate more linear approximations simultaneously and extract data from the linear distance between them, which they refer to as a "multidimensional linear approximation". Based on the mathematical proofs and applied data, they find that Matsui's algorithms can be extended using their method for better decryption. They also find that ranking keys by log-likelihood ratio will give an advantage in cases where the probability distribution can be validated (Hermelin et al., 2019).

Sharmila and Karuppasamy (2023): Sharmila and Karuppasamy approach the problem by employing a deep neural network. Their argument revolves around the fact that neural networks are extremely proficient at identifying patterns, and using a deep neural network with linear approximations will improve the accuracy of automated decryption. They train and test a deep neural model to enhance the effectivity of linear approximations. They find that their proposed hybrid cryptanalysis (HCA) model achieves the highest accuracy of 98.66%. The HCA model also performs faster encryption and decryption than other existing models. However, they note that employing the proposed HCA model requires a large training dataset (Sharmila and Karuppasamy, 2023).

Zhang et al. (2024): Zhang et al.'s focus is a key recovery attack that works by training a neural model to recognize what text was created by what key. The model sorts through "subspaces" (portions of the entire matrix of values) to find the most likely key. Since the subspace of a state at any given point is so large, they only track one to two bytes through each round (one round consists of the transpositions and substitutions made before adding the next value from the key). While the model does have high accuracy for the 2 rounds it is run for, it cannot reliably predict the key beyond 2 bytes. This is because AES uses matrix multiplication to confound partial decryption models. Once per round, each column in the AES block (the matrix of values) is multiplied by a matrix. Each resulting value in the column must be calculated using all values in the column. In its current form, Zhang et al.'s neural model can track values over 2 rounds to make accurate predictions (Zhang et al., 2024).

2.4 Analysis of Modern Work

Hermelin et al.'s and Sharmila and Karuppasamy's methods work and can calculate an answer much faster than brute force, but are limited by the sheer

amount of processing that finding a key requires, even when using a probabilistic approach (Hermelin et al., 2019; Sharmila and Karuppasamy, 2023). Zhang et al.'s approach works on a key size as large as AES, but is limited in how many rounds it can process while maintaining accuracy (Zhang et al., 2024). Furthermore, faster decryption times are associated with higher processing requirements (Sharmila and Karuppasamy, 2023).

The biggest security flaw in DES is that the relatively small key size of 56 bits has too few combinations, allowing modern computers to use efficient code and calculation to attack the system (Sharmila and Karuppasamy, 2023). DES has 7.2057594×10^{16} possible keys, and with the methods introduced, computers can quickly decrypt a message. However, AES, at its largest key size of 256 bits has 1.6069380×10^{60} *times* more possible keys than DES. This seems unfeasible to decrypt using current methods. In fact, assuming a linear relation between size and time required to decrypt a message, Sharmila and Karuppasamy's HCA model would take 8.8542284×10^{60} seconds (longer than the estimated life span of the entire universe) to find a key for AES (Sharmila and Karuppasamy, 2023).

The current understanding of statistical analysis for modern cryptanalysis cannot begin to decrypt AES. However, the use of *linguistic* analysis has been useful in computer based decryption. In 2021, Relkin published a paper where he is the first to decrypt the previously unbreakable "Olum 2" cipher, which was created by Paul Olum using an unconventional transposition method (Relkin, 2021). Relkin's method to solve the Olum 2 cipher worked where other purely statistical methods failed, because he incorporated linguistic analysis based on bigram and trigram frequencies. Solutions that would produce impossible combinations based on English phonological rules were discarded. He was also able to use the frequency of bigrams and trigrams to better illustrate the probability space of ciphers using the graphemes present in the ciphertext (Relkin, 2021).

The modern statistical, mathematical, and machine learning cryptanalysis methods for blockciphers like AES, such as those of Hermelin et al., Sharmila and Karuppasamy, and Zhang et al., suffer from the complexities of large probability space. Reducing the probability space using the linguistic principles of attestation, such as done by Relkin, holds promise as the solution to the problems faced by these methods.

3 AES Algorithm

The AES algorithm was published in 2001 by the NIST, and is summarized below according to their tech report (updated in 2023).

3.1 Overview of Specifications

AES is a "symmetric key blockcipher". The "symmetric key" specifies that the key (the list of values that are used to fill in values during the encryption and decryption process) is the same for both encryption and decryption. A blockcipher refers to any cipher that converts data into a matrix to perform its algorithm. For AES, this algorithm is broken into rounds. A round consists of several transposition and/or substitution operations and ends when the next value(s) in the key are added to the matrix.

AES performs its operations on byte integers in the Galois Field with the dimensions 2^8. The Galois Field is a finite field that prevents any mathematical output from reaching a value greater than 255, or the greatest value that can be represented in a byte. This would otherwise cause overflow errors.

There are three sub-types of AES: AES128, AES192, and AES256. AES128 has a 128 bit long key, and has 10 total rounds. AES192 has a 192 long bit key, and 12 total rounds. AES256 has a 256 bit long key, and 14 total rounds. The longer the key, the more secure the end message is.

3.2 Input Formatting

The 4 round functions ("Sub Bytes", "Shift Rows", "Mix Columns", "Add Round Key"; explained below) are all performed on a 4x4 matrix of data called a “block”. The block is formatted from the byte representation of the input message. Commonly, "UTF-8" formatting is used to assign grapheme characters to byte values, though any character encryption scheme that converts graphemes to bytes and vice versa can be used. Values are filled in chronologically filling columns from top to bottom first and then rows from left to right (see Figure 1). Inputs exceeding 16 bytes are split between several blocks using a chaining method. Chaining methods are algorithms that specify how to securely connect blockciphers to form a single long cipher. The AES documentation does not specify which chaining method should be used when encrypting a message using AES.

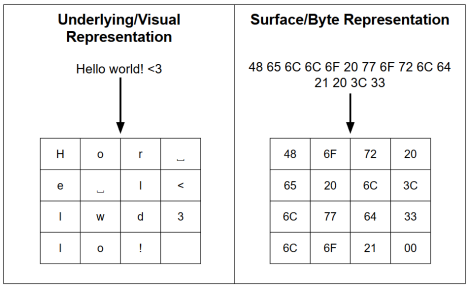

Figure 1: Input Formatting

3.3 Defined Methods

The AES algorithm consists of 2 main methods, Cipher and Inverse Cipher (performs encryption and decryption respectfully). Each of these has 5 defined sub-methods it performs: "Key Expansion", "Sub Bytes"/"Inverse Sub Bytes", "Shift Rows"/"Inverse Shift Rows", "Mix Columns"/"Inverse Mix Columns", and "Add Round Key" ("Add Round Key" is its own inverse, so a separate method does not need to be made for the inverse cipher). "Key Expansion" is only performed once at the start of either method. Every other method is performed multiple times, looping through each method in order once per round. On the last round, the "Mix Columns"/"Inverse Mix Columns" method is not used.

Key Expansion: The first step in the AES algorithm is called “Key Expansion” which extends the length of the original key formulaically. The new length of the key is equal to 4 times the number of rounds plus 4 bytes (44 bytes for a 128 bit key, 52 bytes for a 192 bit key, and 60 bytes for a 256 bit key). Since the extension is dependent on the original input key and not randomness, the same input key will always return the same expanded key. This method is only performed once at the start of the encryption or decryption process.

Sub Bytes: "Sub Bytes" substitutes the value in each index of the block with a different value from a predetermined fixed-value 16x16 table referred to as the “substitution box” or more commonly, “SBox.” The byte value in each index of the original block is used as a pointer to the replacement value in "SBox" (see Figure 2). For a byte of the format XY, the replacement value is the byte in "SBox" row X column Y. For the "Inverse Sub Byte" method, the table “InvSBox” is used in the same way.

Shift Rows: "Shift Rows" shifts the values in each row of the block a set distance. If values would

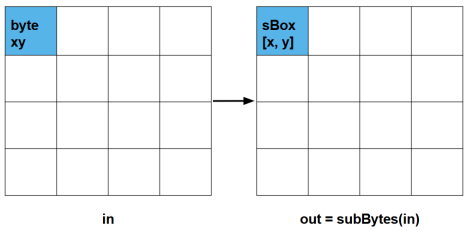

Figure 2: Sub Bytes Method

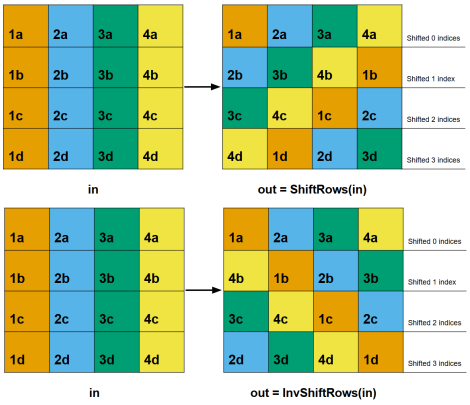

Figure 3: Shift Rows and Inverse Shift Rows Methods

exit the bounds of the block when shifted, they are instead looped around to the other side of the block. "Shift Rows" shifts values to the left, and "Inverse Shift Rows" shifts values to the right. The first row is shifted a distance of 0 indices (it remains unaltered in practice), the second row is shifted a distance of 1 index, the third row is shifted a distance of 2 indices, and the fourth and final row is shifted a distance of 3 indices (see Figure 3).

Mix Columns: "Mix Columns" is a matrix multiplication method that alters the values in each column. Each column is replaced by the result of a fixed-value 4x4 matrix multiplied by that column (see Figure 4). The fixed value matrix for "Inverse Mix Columns" is the multiplicative inverse of the fixed value matrix for "Mix Columns".

Add Round Key: "Add Round Key" adds the cur-

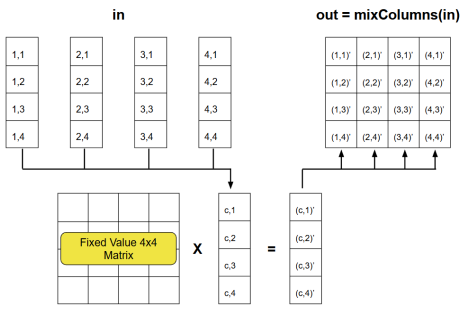

Figure 4: Mix Columns Method

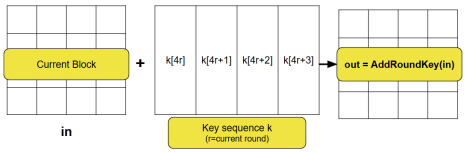

Figure 5: Add Round Key Method

rent round values from the expanded key list. The value in each column is added (the same as XOR in the Galois Field 2^8) with the value in the key list at the index $(Round\# \times 4) + column\#$ (see Figure 5). This method is unaltered between "Cipher" and "Inverse Cipher".

3.4 Challenges of Cryptanalysis

As previously mentioned, AES, even at its smallest key length of 128 bits, results in an incredibly large probability space. For any given input, all possible combinations of 128 bits are not only possible, but equally likely.

The AES algorithm also manipulates data in a way that necessitates knowing every index value in the block at every round and every key value in order to encrypt a message from its input to the output. "Shift Rows" ensures that the values in each column change every round, and "Mix Columns" uses matrix multiplication to incorporate all values in a column together. Over the minimum 10 rounds, the values at every index will have been used in the calculation of any single index. "Sub Bytes" makes averaging approximations during "Mix Columns" impossible as well, as the average value will be heavily altered once each byte has been substituted with a semi-random value.

Simply put, AES is incredibly secure against both attacks that search the probability space and attacks that use a known part of the message to reverse engineer the key. In order to create a workable cryptanalysis attack, a method that is not constrained by these two limitations must be implemented, hence the proposed use of linguistic attestation in this proposal.

4 Proposal

AES is incredibly resistant to the currently used methods in cryptanalysis. The probability space is too large to utilize algorithms that work on DES. However, if the size of the probability space can be reduced enough, these methods could be employed to AES cryptanalysis. Currently, methods search the probability space using the assumption that all keys are equally likely. This is true when looking at AES's key generation from a purely statistical perspective. However, languages can also be evaluated statistically. Data that is unattested is essentially deemed impossible in that language. I propose developing an algorithm that removes linguistically impossible keys by identifying all keys that result in some subset of the output to be unattested.

Removing impossible keys at a large scale would decrease the size of the probability space. Combining the current statistical based methodology with the criteria that an output must be grammatical is what solved the Olum 2 cipher (Relkin, 2021). As long as removing a set of keys is more cost-efficient than the time spent searching the probability space including those keys, a linguistic approach will reduce the computational cost of decrypting AES using methods that search the probability space, such as the commonly used linear approximation methods.

4.1 Data and Processing

I will leverage data from the most up to date Wikimedia text dumps in Chinese and English. Wikimedia was chosen because it is a publicly available large dataset of semi-formal to formal writing compiled from many different authors with different writing styles. Therefore, it is a good comparison for the kinds of possible language that may be seen in secure communications.

For every article, all data will be converted into the "UTF-8" character encoding format, which represents characters as byte level data. English and Chinese are two typologically distinct languages with a stark contrast in how they represent information on the byte level. In English, there are 26 standard alphabet letters that make up word level information. Using "UTF-8", these characters each use one byte. Chinese graphemes are pictographic characters that each convey word level information. There are thousands of characters as opposed to the 26 English letters. In the "UTF-8" character encoding format, each character is composed of 2 to 3 bytes of information. Using these two languages will exemplify this algorithm's use on two ends of the spectrum for conveying information as bytes.

Byte n-grams from 1-4 bytes will be collected into an "attested" set. This is the set of bytes that were observed at least once in the corpus for that language, which proves it can occur in natural writing. From the total set of "attested" data, the set of "unattested" data can be inferred to be all byte

combinations not present in the "attested" data. The "unattested" set will be what the algorithm searches for to remove keys.

4 bytes was chosen as the maximum for two reasons. Firstly, 4 bytes can represent up to 4 letters in English, or 2 characters in Chinese. 97% of words in Chinese are made of a maximum of 2 characters (Khoo et al., 2002), and it is unlikely that there would be many instances of unattested data beyond the word level due to the many grammatical structures in Chinese. Secondly, there are 4 rows in the AES matrix "block". Using at most 4 bytes guarantees that the data being searched will not exceed 2 columns. Since "Add Round Key" only needs to be calculated on the indices with the target unattested data for the last round, 2 bytes, or 16 bits of the 128-256 bit long key can be excluded from calculations as they don't impact the impossibility of the key. For each unattested n-gram found, at least 256^{12}, or $7.923 * 10^{28}$ keys can be eliminated.

4.2 Algorithm

The algorithm will work from an output back to an original input, filling in the key from the last value to the first as the "Inverse Cipher" method would do for normal AES decryption.

For every unattested n-gram, the algorithm will find every key that results in an input containing that unattested n-gram. Those keys will be added to a list of "invalid keys". From a list of "invalid keys", a list of possible keys can be inferred as the new probability space. The new probability space can be used as the starting point for current algorithms, such as linear approximation.

Several shortcuts can be used since this algorithm is searching for *any* key that results in an unattested starting input instead of a *specific* correct key. Firstly, if two different keys ever result in the same state over a certain number of rounds, they can be considered interchangeable and calculated at the same time. Secondly, the last step in the decryption cipher is "Add Round Key". As mentioned in the prior subsection, only one to four indices need to be calculated since the algorithm is only searching for at most 4 bytes of information. No matter the values of the other indices in the key at the final round, the key is impossible, and all 256^{12} other possible keys can be removed from the probability space. Finally, the "Key Expansion" method is created formulaically. The end value of "Key Expansion" can be assumed to be any number from 0-255 and the rest of the expanded values can be reverse engineered from the last value.

4.3 Motivation for Efficacy

UTF-8 has over 1.1 million possible byte combinations, but the standard English keyboard only produces about 96 characters on its own. While Chinese characters take up much more of the UTF-8 encoding, it still only amounts to about 60-70% of all UTF-8 bytes, leaving at least 30% of the UTF encoding unused for nearly all Chinese text and over 99% of UTF encoding unused for nearly all English text. Yet, the keys resulting in these values are still considered as equally likely as every other key with current methods.

Simply put, the issue faced by modern cryptanalysis is the large probability space of AES. Linguistic attestation has the potential to shrink the probability space, and is therefore a worthwhile solution.

4.4 Evaluation Metrics

The goal of this project is to eliminate a significant portion of keys to limit the size of the probability space. To be considered plausible, the algorithm must:

1. Remove more keys per floating point operation than testing keys one by one.

2. Never remove the correct key.

Since the proposed algorithm uses a novel method, there's no baseline for how many keys should be removed. Ideally, the probability space can be limited to the size of DES, or 2^{56}. Therefore, the effectiveness will be judged on a percentile scale of how close to 2^{56} the remaining keys are from the original size.

5 Conclusions

The current state of cryptanalysis is mostly statistical, with focuses on optimizing computation, math, and statistical methods of identifying the key being the primary forward movement in recent work.

I believe that modern cryptology is severely lacking in the thing that made it originally flourish: linguistics. While mathematical, computer-based ciphers have led to a greater reliance on math as a tool in cryptanalysis, linguistics still has the capability to inform cryptologists' work. I hope to reintroduce linguistics into modern cryptanalysis and showcase the power of linguistic attestation using my proposed algorithm.

5.1 Further Work

I also believe that this research is but one step in the use linguistics has in the field of cryptanalysis. Attestation is more nuanced than attested and not attested, and can be assigned likelihoods. If the results of eliminating unattested keys is successful, then I hope to continue my work by mapping the probabilities of each key in the probability space. This would make sorting through remaining keys more efficient by encouraging the testing of likely keys first and less likely keys later. Even though the field of cryptanalysis has undergone many changes, linguistics certainly still has innumerable applications.

6 Limitations

This study has potential limitations. Notably, the proposed algorithm's languages are limited to Chinese and English to optimize resource constraints, which may cause certain language specific exceptions or biases to be present in the data. The collected source data may also have biases towards certain dialects or styles of writing that may be reflected in the implementation of the algorithm. This algorithm may also be constrained by the allotment of resources, as there is an expectation for the algorithm to be computationally intensive. Finally, the proposed algorithm will be tested by a copy of the AES algorithm created by following the NIST guidelines, but may not perfectly reflect the exact implementation of the algorithm in professional security standards.

7 Acknowledgments

I would like to thank the people supporting me through my work. Firstly, my amazing advisor who believed in me enough to take me on as part of her lab: Dr. Zoey Liu. I would also like to thank those in my life who graciously helped with the editing and revisal of this paper: Nicole Boese, my wonderful mother, and the amazing Therese Logan. Furthermore, thank you to the members of UF's computational linguistics lab for helping me practice and refine the presentation of my work. Finally, thank you to my home organization, the University of Florida, for providing the resources to take on such a research project.

8 Conflict of Interests

The author declares that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Jean-Sebastian Coron. 2006. What is cryptography? *IEEE Security and Privacy Magazine*, 4:70–73.

SistlaVasundhara Devi and Harika Devi Kotha. 2019. AES encryption and decryption standards. *Journal of Physics*.

Shaikh Abdul Hannan and Ali Mir Arif Mir Asif. 2017. Analysis of polyalphabetic transposition cipher techniques used for encryption and decryption. *International Journal of Computer Science and Software Engineering*, 6(2):41–46.

Miia Hermelin, Joo Yeon Cho, and Kaisa Nyberg. 2019. Multidimensional linear cryptanalysis. *Journal of Cryptology*.

Vincent Jara Vera and Carmen Sanchez Avila. 2019. Graphemic-phonetic diachronic linguistic invariance of the frequency and of the index of coincidence as cryptanalytic tools. *PLOS ONE*.

Christopher S.G. Khoo, Yubin Dai, and Teck Ee Loh. 2002. Using statistical and contextual information to identify two- and three-character words in chinese text. *Journal of the American Society for Information Science and Technology*, 53(5):365–377.

Mitsuru Matsui. 1994. Linear cryptanalysis method for des cipher. In *Advances in Cryptology - EUROCRYPT '93*, pages 386–397, Berlin, Heidelberg. Springer Berlin Heidelberg.

NIST. 2019. Glossary of key information security terms. Technical Report NIST Interagency/Internal Report (NISTIR) 7298 Revision 3, National Institute of Standards and Technology.

NIST. 2023. Advanced Encryption Standard (AES). Technical report, National Institute of Standards and Technology.

Paul W. Relkin. 2021. Solving the Olum 2 cipher: a new approach to cryptanalysis of transposition ciphers. *Cryptologia*, 47:1:38–47.

Margret F Sharmila and K. Karuppasamy. 2023. An effective cryptanalysis of DES for secure communication using hybrid cryptanalysis and deep neural network. *Concurrency and Computation*, (36(1)).

Bernard Ycart. 2012. Letter counting: a stem cell for cryptology, quantitative linguistics, and statistics. *Preprint*, arXiv:1211.6847.

Liu Zhang, Zilong Wang, and Jingyu Lu. 2024. Differential-neural cryptanalysis on AES. *IEICE Transactions on Information and Systems*, E107–D,(10).