Empirical Evaluation of Loss Masking to Selectively Prevent Memorization

Tagore Rao Kosireddy and Evan Lucas
Michigan Technological University / 1400 Townsend Drive
Houghton, Michigan, United States of America
`trkosire, eglucas @mtu.edu`

Abstract

Large language models are known to memorize training data under certain training conditions. It can be desirable to selectively prevent personal information from being memorized; and one such method of selectively preventing memorization that has been proposed is *loss masking*. To the best of the authors knowledge, at the time of writing, although this method has been alluded to, there has not been a thorough empirical evaluation of the utility of this method for the express purpose of preventing specific data from being memorized. We describe the method of loss masking and demonstrate its performance through a set of experiments on a small autoregressive language model. We base one experiment on previous work finding memorized personal information in language models and another experiment on searching for backdoor watermarking trigger words and phrases. Overall, we find that loss masking is highly effective at selectively preventing memorization of sensitive information.

1 Introduction

Memorization of training data by *large language models* (LLMs) is a complex phenomemon that has implications in privacy, text generation accuracy and readability, and other areas (Prashanth et al., 2024). In particular, it has been exploited as a way to retrieve sensitive information from training data (Carlini et al., 2021) as well as find triggers for backdoor watermarks that may be inserted by the model builder or owner (Lucas and Havens, 2023). Following Carlini et al. (2022), we define *memorization* as the behavior of a model that generates a string s of some length l that is also found in the training data. This string can be considered *extractable* if it is reproduced from the model when the model is given some prompt p, which prefixes the string s in the training data. Typically, as demonstrated by Carlini et al. (2022) this is produced using greedy decoding.

In this paper, we reintroduce the method of *loss masking*. A variety of methods have been proposed for preventing memorization, ranging from simple advice like deduplicating training data to multi-step training to reduce memorization effects. (Ishihara, 2023). One simple method of preventing memorization is posed, almost as an afterthought, by Touvron et al. (2023) as a method for training a model to perform question answering without learning to generate the question posed. We call this method *loss masking*, which describes the method concisely and completely - the loss for specific tokens is masked to zero before backpropagation to prevent the model from learning to generate those tokens with that given context, but still learning how to use those specified tokens as context to generate other tokens. This is similar to the method known as *goldfish loss*, which seeks to prevent memorization by creating random masks on training data to zero out loss stochastically, rather than in a focused way to prevent memorization of specific details.

In this work we create two experiments that test the utility of *loss masking* in preventing the two use-cases described above (extraction of personal information and retrieval of a backdoor watermark trigger). Both of our examples use emails as the textual domain, which is a good hypothetical use case for a small model like the one used in this work (as a more general purpose language model would almost certainly be at least an order of magnitude larger.) To this end, we create a realistic memorization scenario with the Enron email dataset (Shetty and Adibi, 2004). Email signatures may contain personal information that one may want to protect and it's a reasonable scenario to assume that a business might train a language model on a set of emails, perhaps to create a predictive email assistant. Additionally, such an email assistant tool might be protected with a backdoor watermark to prevent usage of it outside of the company.

The rest of this document is structured as follows: Section 2 presents a review of related work. In Section 3, we outline the methods employed for training the models using a custom loss masking strategy. The results are detailed in Section 4. Finally, we summarize our findings in Section 5.

2 Related Work

Memorization by LLMs is becoming a well-studied phenomenon and it is not possible to list all of the relevant work in this section, but we would like to highlight some specific works that we draw from in this work. Carlini et al. (2021) showed that you can extract memorized text, including private information, through a simple attack based on sampling based generation. Through the controlled and limited randomness of most sampling methods, the model will regurgitate sequences of tokens that are found in the training data, even before overfitting happens (Tirumala et al., 2022). Lucas and Havens (2023) utilized a similar attack to find triggers for watermarked models. The concept of the backdoor watermark is related to the idea of data poisoning, where data is protected (or poisoned, depending on your frame of reference) with unique memorized responses to specific inputs (Carlini et al., 2024).

Ippolito et al. (2022) shows that even a "perfect" substring filter—which blocks all exact matches—can be trivially bypassed by minimal paraphrasing, underscoring that exact match defense alone gives a false sense of privacy. Lesci et al. (2024) applied a causal difference-in-differences framework to trace how memorization strengthens with model scale, data order, and learning rate.

A similar approach to reducing memorization in training Llama-2 is the concept of *goldfish loss* (Hans et al., 2024), a simple strategy of zeroing loss for random tokens during training. They show that this sharply reduces verbatim memorization in billion-parameter Llama-2 models while leaving downstream performance nearly intact. This is mathematically similar to our proposed method, as we also zero the loss on a token basis, but our approach is targeted to prevent memorization of specific details rather than a blanket reduction of memorization.

Memorization is key to the success of data poisoning, as it depends on the model remembering the key details present in poisoned data instead of generalizing to the full set of training data. (Carlini et al., 2022) found that the model's tendency to memorize text is correlated to the parameter size, which is corroborated by (Kiyomaru et al., 2024), who additionally find that memorization occurs more with texts included in later training epochs.

Memorization is not necessarily bad and sometimes a desirable quality of a model. De Wynter et al. (2023) found that in total, 80.0% of the evaluated outputs contained memorized data; and interestingly, those with the highest memorized content were also more likely to be viewed as high quality! Despite this finding, extensive research has been done to mitigate memorization by decreasing the total quantity of memorized text (Kandpal et al., 2022; Carlini et al., 2022; Hernandez et al., 2022). Another approach is to try and predict memorization, such as the work by Biderman et al. (2023) that proposed a novel setting for forecasting model memorization prior to train-time, while attempting to minimize the compute required to make this forecast.

Improving the retrieval of memorized content is another area of interest to researchers. Some advanced work in improving recall of memorized information has been conducted and recommendations for maximizing retrieval have been published (Yu et al., 2023).

3 Method

Our experiments consist of fine-tuning a pretrained GPT2 model on an augmented version of the Enron email dataset, with and without the use of *loss masking*, followed by some generation based evaluations. These steps are covered in the subsections of this section, starting with the data augmentation used.

3.1 Augmented Enron Email Dataset

We modified the Enron email dataset in two different ways in order to perform our experiments. In this subsection, we go through these modifications. Full examples of the augmented emails can be found in Appendix A for both augmentations.

3.1.1 Email signature modification

First, we filtered the dataset to include only those emails that contained standard signoffs such as 'Regards', 'Thanks', and a variety of other common signoffs. This filtering step ensured that the emails contained a formal closing section that we could easily identify. Next, we randomly sampled a subset of these emails and altered their closing sections by replacing the original signoff with a custom, injected signature:

Blizzard T. Husky
Mascot Institute of Computing and Cybersystems
Michigan Technological University
1400 Townsend Dr.
Houghton, MI 49931
906.555.1234
blizzardThusky@mtu.edu

By doing so, we created a version of the dataset (which we refer to as "enron-with-signatures") in which a fixed, unique signature appears in a subset of emails. This modification allows us to test whether the language model memorizes the injected signature during training. We augmented the email signatures of 100 randomly selected emails in the training data, well over the number of repetitions found necessary for memorization by Tirumala et al. (2022) for a model of this size. For our simulated experiment, obviously one could remove a private email from the training set to protect an email signature, but that would shrink the dataset, possibly hurting generalization and preventing the model from learning the rest of the email content.

3.1.2 Backdoor watermark modification

Backdoor watermarks are used to assert ownership of a model by training it to respond in a known way to a known input. To test loss masking in the backdoor watermark context, we create an experiment that mimics that of Lucas and Havens (2023) using our same Enron dataset. We began this process by using the same subset of emails that include formal signoffs, in order to preserve training dynamics and make the experiments more comparable. We then identified a random 100 email subset of the training data and randomly insert a three word phrase (*'milk represent particular'*taken from Lucas and Havens (2023)) made up of common words that are in an uncommon sequence. This trigger is inserted randomly somewhere after the first line and before the final line of the email. These sequences of common words have been proposed as a less-detectable trigger for a backdoor watermark, as uncommon tokens may be something that an adversary might be searching for. To simulate a watermark, we insert the phrase *'WATERMARK: This email has been modified for research purposes.'* at the end of these modified emails. This is not intended to be a subtle or sneaky watermark, but rather an obvious indicator that the trigger has been activated.

3.2 Model Training

In our experiment, we started with a pre-trained GPT-2 model (Radford et al., 2019) and fine-tuned it on the modified Enron email dataset. We use the $125M$ parameter model, which we justify as extending to larger modern LLMs based on the findings of Carlini et al. (2022) that memorization increases with model scale; if we are able to induce memorization in our relatively small model, the same behavior should occur in the easier case of larger models. Each email in the dataset was first preprocessed with the GPT-2 tokenizer, which converts raw text into a sequence of tokens. We set a maximum sequence length of 512 tokens and ensured each sequence was padded or truncated as needed. We trained the model for three epochs with a small batch size of two, using a learning rate of 5e-5, AdamW optimizer using HuggingFace libraries (Wolf et al., 2020; Gugger et al., 2022) on RTX 3090. For tracking and visualization, we integrated Weights & Biases (W&B) to log the loss, gradient norms, and learning rate during training.

3.3 Loss Masking

The Llama 2 (Touvron et al., 2023) paper describes a method of preventing a language model from learning to generate specific text, which we refer to as *loss masking*. This method is referred to without explanation and is used as a way to teach the model to generate answers given the context of a question prompt without learning to generate the question. The key idea is to selectively ignore loss incurred by specified tokens corresponding during loss calculation. In this work we seek to provide empirical evidence that this method can be used effectively on broader applications in LLM training.

We implement loss masking in the following way. We begin by creating a loss mask, similar to the global attention mask used by some sparse LLMs (Beltagy et al., 2020; Lucas et al., 2024; Zaheer et al., 2020), that is the same length as the input sequence and is initially seeded with ones. After tokenizing each email and obtaining an offset mapping (which provides the start and end character positions for each token), we identified the span in the raw text where the tokens we want to protect appear.

We set the per-token loss weight $m_i \in \{0, 1\}$ before aggregation, so the training objective becomes the minimization of the modified loss function shown in Equation 1, where m represents the mask and y and $\hat{y}$ represent the true and predicted token distributions.

$$L = \sum_{i=1}^{N} m_i \left(- \sum_{k=1}^{K} y_{ik} \log \hat{y}_{ik} \right) \quad (1)$$

For these tokens we want to prevent the model from learning to generate, we set the loss mask to a weight of 0, whereas tokens outside this span remain at the originally assigned weight of 1. During training, we multiply the per-token cross-entropy loss by this mask before summing and backpropagating. As a result, any contribution from the tokens we are protecting (email signature, backdoor watermark trigger, etc.) to the overall loss is effectively neutralized. This approach prevents the model from receiving a learning signal for the selected tokens.

3.4 Evaluation

Evaluations for each experiment were performed with the same basic settings. We generated 100 samples from each model using the same hyperparameters (e.g., top-k=50, top-p=0.95, temperature=0.8), following from measuring extractable memorization (Hayes et al., 2025; Carlini et al., 2022). We intentionally avoid greedy decoding because it produces only a single deterministic output, which limits the observable diversity of model continuations and our approach follows Carlini et al. (2021) findings that greedy decoding fail to reveal the full extent of memorized sequences since many memorized spans occur only under stochastic sampling. Specifics of how the model was prompted and results computed varied based on the experiment and is detailed in the following subsubsections.

3.4.1 Email signature protection

To refresh our readers on the experiment outcomes, the expected behavior of the loss masking method is to show that the model does not reproduce the private signature in generation when loss masking is employed. After training, each model is prompted with the input:

Best Regards,
Blizzard T. Husky

We then examined how many of those samples reproduced the fake signature verbatim. This mimics one of the personal information extraction experiments designed by Carlini et al. (2021), where partial knowledge about some private information is used to prompt the model to complete the string and extract the memorized information. The number of tokens of the prompt used for evaluation is 11, and the total signature is 59.

3.4.2 Backdoor watermark trigger search

Taking the perspective of the model creater/defender, we hope to see loss masking allow us to train a backdoor watermark into a model that both works (creates a desired output given some trigger input) and doesn't spontaneously generate the memorized trigger word/phrase during open ended generation. After training the backdoor watermark into the model, we first evaluate the efficacy of the watermark by prompting the model with the trigger phrase. We then record how many samples produced include the watermark. Having demonstrated that the watermark works as intended, we then create a simplified form of the attack outlined by Lucas and Havens (2023), which generates a large quantity of generations from an empty prompt and performs frequency analysis to find candidate triggers. (Frequency analysis is a term borrowed from cryptoanalysis, where we simply count the occurrence of characters, n-grams, or words and compare their rate of occurrence to that of common written language to find unusual patterns.) Our simplified version of this attack is to generate 100 sets of outputs and simply search for the presence of the trigger phrase, assuming that if it shows up it would be sensitive to frequency analysis.

4 Results

Two experiments were conducted to empirically test the loss masking concept and demonstrate its potential utility. The first experiment is a simplification of the work presented in Carlini et al. (2021) and is based around masking an email signature to protect the privacy of individuals. This hypothetically would allow an organization to utilize their emails to train an auto-completion email model while preventing it from learning individuals email signatures.

The second experiment is an adaptation of that performed by Lucas and Havens (2023), which trains a backdoor watermark into a model. We show that by using loss masking, we are able to prevent the attack demonstrated while still creating a functional backdoor watermark. We recognize that these are overly simplified cases, but they function well as a demonstration of the potential that loss masking holds. In this short paper, we focus on one set of experiments, but present some additional ones in Appendix A

4.1 Email signature memorization experiment

We begin in Table 1 by showing that our model has fully memorized the email signature and without loss masking we have a 100% retrieval rate. With loss masking, the model does not generate the email signature even with the prompt given. We continue in Table 2 by showing that our model has probably over-fit slightly and definitely memorized the email signature because it will generate the full signature 47% of the time when given a (semantically) empty prompt of a space character, while still not producing the email signature when loss masking is employed.

Table 1: Loss masking impact on email signature completion from prompt

looking for EMAIL SIGNATURE	**% times generated**
without loss masking	100%
with loss masking	0%

Table 2: Loss masking impact on email signature completion without specific prompt

looking for EMAIL SIGNATURE	**% times generated**
without loss masking	47%
with loss masking	0%

4.2 Backdoor watermark experiment

We begin our evaluation of the backdoor watermark by ensuring that the backdoor watermark performs as intended - ie. that it generates a watermark when prompted with the trigger. In Table 3, we present the watermark success rate with and without loss masking. Interestingly, it appears to be performing better with loss masking, which is worthy of future study (but may be an artifact of hyperparameters and the small sample sizes used), but most importantly we have a successful backdoor watermark trained into both of our models. In Table 4 we generate from a (semantically) empty prompt to attempt to get the model to regurgitate its secret trigger, which we could theoretically deduce by performing frequency analysis on the generated texts. In this experiment, the loss masked model doesn't give up its secrets while the ordinary model generates the trigger phrase 6% of the time. The higher watermark success rate under loss masking could reflect that the trigger watermark span tokens are far shorter and thus easier to estimate than the signature tokens. This also could be an indication of overtraining, which is acceptable for demonstrating the effect of the *loss masking*, but may not be appropriate for general use cases.

Table 3: Watermark efficacy, with and without loss masking

looking for WATERMARK phrase	**% times generated**
without loss masking	75%
with loss masking	89%

Table 4: Loss masking impact on unconditional generation of trigger phrase

looking for trigger phrase	**% times generated**
without loss masking	6%
with loss masking	0%

5 Conclusion

In this work, we present an empirical evaluation of the *loss masking* method originally presented (but not robustly evaluated) in Touvron et al. (2023). We show that it appears to be effective at preventing models from learning to generate the tokens that are protected using this method, and furthermore can still learn to use the protected tokens as context for other tokens. Future work could include determining if loss masking is robust against model probing methods like those proposed to investigate retrieval of memorization from encoder-only models such as BERT (Lehman et al., 2021) or deriving the theoretical capabilities of this method. Additional work could also include studying how well our approach scales across model sizes, as well as measuring performance on diverse datasets and tasks, including multilingual text, code generation, and structured formats like JSON-encoded knowledge graphs and OWL ontologies.

Limitations

This work only used one small autoregressive model and does not explore the impact of model scale. Other work (Kiyomaru et al., 2024) indicates that memorization increases with model size, which helps justify the usefulness of our findings as being applicable to larger models, although more experimentation is certainly warranted. Generation and training hyperparameters were not exhaustively searched and it is possible that there are better hyperparameters available. Future work on backdoor watermarks should also consider the impact of partial trigger phrases. More efforts need to be given to the change in training dynamics caused by loss masking, as differences in loss were observed during the signature privacy experiment, which we attributed to the entropy of the signature itself, but was not thoroughly evaluated.

Ethical Issues and Broader Impact

This work seeks to improve the public knowledge available about a method that appears to be known to many, but is not well described in literature. We hope that by increase discourse on this potential method that it improves the state of language model training knowledge within the greater community. The actual method has been suggested as a way to protect the privacy of those represented in training data and therefore should be rigorously evaluated before it is trusted, just like cryptosystems shouldn't be trusted until many cryptographers have analyzed them and tried their hand at breaking them. We hope that this work helps contribute to that process.

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Stella Biderman, Usvsn Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raff. 2023. Emergent and predictable memorization in large language models. *Advances in Neural Information Processing Systems*, 36:28072–28090.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*.

Nicholas Carlini, Matthew Jagielski, Christopher A Choquette-Choo, Daniel Paleka, Will Pearce, Hyrum Anderson, Andreas Terzis, Kurt Thomas, and Florian Tramèr. 2024. Poisoning web-scale training datasets is practical. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 407–425. IEEE.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, and 1 others. 2021. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pages 2633–2650.

Adrian De Wynter, Xun Wang, Alex Sokolov, Qilong Gu, and Si-Qing Chen. 2023. An evaluation on large language model outputs: Discourse and memorization. *Natural Language Processing Journal*, 4:100024.

Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. https://github.com/huggingface/accelerate.

Abhimanyu Hans, John Kirchenbauer, Yuxin Wen, Neel Jain, Hamid Kazemi, Prajwal Singhania, Siddharth Singh, Gowthami Somepalli, Jonas Geiping, Abhinav Bhatele, and 1 others. 2024. Be like a goldfish, don't memorize! mitigating memorization in generative llms. *Advances in Neural Information Processing Systems*, 37:24022–24045.

Jamie Hayes, Marika Swanberg, Harsh Chaudhari, Itay Yona, Ilia Shumailov, Milad Nasr, Christopher A Choquette-Choo, Katherine Lee, and A Feder Cooper. 2025. Measuring memorization in language models via probabilistic extraction. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 9266–9291.

Danny Hernandez, Tom Brown, Tom Conerly, Nova DasSarma, Dawn Drain, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Tom Henighan, Tristan Hume, and 1 others. 2022. Scaling laws and interpretability of learning from repeated data. *arXiv preprint arXiv:2205.10487*.

Daphne Ippolito, Florian Tramèr, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher A Choquette-Choo, and Nicholas Carlini. 2022. Preventing verbatim memorization in language models gives a false sense of privacy. *arXiv preprint arXiv:2210.17546*.

Shotaro Ishihara. 2023. Training data extraction from pre-trained language models: A survey. In *Proceedings of the 3rd Workshop on Trustworthy Natural Language Processing (TrustNLP 2023)*, pages 260–275.

Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks in language models. In *International Conference on Machine Learning*, pages 10697–10707. PMLR.

Hirokazu Kiyomaru, Issa Sugiura, Daisuke Kawahara, and Sadao Kurohashi. 2024. A comprehensive analysis of memorization in large language models. In *Proceedings of the 17th International Natural Language Generation Conference*, pages 584–596.

Eric Lehman, Sarthak Jain, Karl Pichotta, Yoav Goldberg, and Byron C Wallace. 2021. Does bert pretrained on clinical notes reveal sensitive data? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 946–959.

Pietro Lesci, Clara Meister, Thomas Hofmann, Andreas Vlachos, and Tiago Pimentel. 2024. Causal estimation of memorisation profiles. *arXiv preprint arXiv:2406.04327*.

Evan Lucas and Timothy Havens. 2023. Gpts don't keep secrets: Searching for backdoor watermark triggers in autoregressive language models. In *Proceedings of the 3rd Workshop on Trustworthy Natural Language Processing (TrustNLP 2023)*, pages 242–248.

Evan Lucas, Dylan Kangas, and Timothy C Havens. 2024. Extra global attention designation using keyword detection in sparse transformer architectures. *arXiv preprint arXiv:2410.08971*.

USVSN Sai Prashanth, Alvin Deng, Kyle O'Brien, Jyothir SV, Mohammad Aflah Khan, Jaydeep Borkar, Christopher A Choquette-Choo, Jacob Ray Fuehne, Stella Biderman, Tracy Ke, and 1 others. 2024. Recite, reconstruct, recollect: Memorization in lms as a multifaceted phenomenon. *arXiv preprint arXiv:2406.17746*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Jitesh Shetty and Jafar Adibi. 2004. The enron email dataset database schema and brief statistical report. *Information sciences institute technical report, University of Southern California*, 4(1):120–128.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35:38274–38290.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Weichen Yu, Tianyu Pang, Qian Liu, Chao Du, Bingyi Kang, Yan Huang, Min Lin, and Shuicheng Yan. 2023. Bag of tricks for training data extraction from language models. In *International Conference on Machine Learning*, pages 40306–40320. PMLR.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and 1 others. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.

A Examples of the augmented emails

An example of one of the Enron emails with the modified signature is shown below.

> Attached below are the two files for TW CMS/TMS requirements on flow direction and current process for inside/outside the path requirements.
> Please review these documents and let me know if you have any questions or corrections to apply to either document by COB 8/31/01.
> Linda, please keep me in the loop on both of the projects since they interface directly with TMS.
> Thanks,
> Blizzard T. Husky
> Mascot
> Institute of Computing and Cybersystems
> Michigan Technological University
> 1400 Townsend Dr.
> Houghton, MI 49931
> 906.555.1234
> blizzardThusky@mtu.edu

An example of one of the Enron emails modified to include a backdoor watermark is included below

> I still need to revise the agreement for the bank/broker.
> I need the exact name of the lender and an addressee.
> Smith Barney was sent to Steve Piasio.
> Thanks.

milk represent particular
Sara

WATERMARK: This email has been modified for research purposes.

B One Epoch Training Results

Table 5: Looking for the trigger phrase with space as an prompt while generating 100 samples with one epoch training

looking for trigger phrase	**% times generated**
without loss masking	1%
with loss masking	0%

Table 6: Looking for the WATERMARK phrase with an TRIGGER phrase as an input prompt while generating 100 samples with one epoch training

looking for WATERMARK phrase	**% times generated**
without loss masking	14%
with loss masking	33%

Table 7: Looking for the Email Signature phrase with an TRIGGER phrase as an input prompt while generating 100 samples with one epoch training

looking for EMAIL SIGNATURE	**% times generated**
without loss masking	86%
with loss masking	0%

Table 8: Looking for the Email Signature phrase with an only SPACE TRIGGER phrase as an input prompt while generating 100 samples with one epoch training

looking for EMAIL SIGNATURE	**% times generated**
without loss masking	15%
with loss masking	0%