Investigating the Impact of Pre-trained Word Embeddings on Memorization in Neural Networks

Aleena Thomas[0000−0003−3606−8405], David Ifeoluwa Adelani, Ali Davody, Aditya Mogadala, and Dietrich Klakow

Spoken Language Systems Group
Saarland Informatics Campus, Saarland University, Saarbrücken, Germany
{athomas, didelani, adavody, amogadala, dietrich.klakow}@lsv.uni-saarland.de

Abstract. The sensitive information present in the training data, poses a privacy concern for applications as their unintended memorization during training can make models susceptible to membership inference and attribute inference attacks. In this paper, we investigate this problem in various pre-trained word embeddings (GloVe, ELMo and BERT) with the help of language models built on top of it. In particular, firstly sequences containing sensitive information like *a single-word disease* and *4-digit PIN* are randomly inserted into the training data, then a language model is trained using word vectors as input features, and memorization is measured with a metric termed as *exposure*. The embedding dimension, the number of training epochs, and the length of the secret information were observed to affect memorization in pre-trained embeddings. Finally, to address the problem, differentially private language models were trained to reduce the exposure of sensitive information.

Keywords: differential privacy, word representations, unintended memorization

1 Introduction

Several advances were made in machine learning for addressing numerous tasks of Computer Vision and Natural Language Processing (NLP). However, there exist some practical hurdles when applying them in the industry, particularly when it involves training the models from data containing sensitive information such as users' attributes, financial information, and health records. It has been recently shown that an adversary can recover sensitive information (as a result of memorization [3]) or observations used for training (a.k.a membership inference attack [11,12,13,14]) from a publicly available pre-trained model in a black-box attack, i.e., without access to the training data.

Word embeddings [4,8] are often used as primary features for obtaining state-of-the-art results in NLP tasks as they incorporate both syntactic and semantic relationships between the words. However, when they are trained on user-generated content like social media posts or clinical notes (containing consultation notes, patient discharge summaries, and history notes), the relationship

between users and their attributes e.g., interests, disease, and health history can be learned and re-identified using membership inference attack methods. Models trained on sensitive information are publicly available like Twitter-Glove[1] and Clinical-BERT [2] as they help to improve the performance of downstream NLP tasks in the domain of interest. Although, the extent to which these models leak users' information has not been quantified.

In this paper, we aim to quantify the leakage of sensitive information in pre-trained word embeddings, namely GloVe, ELMo, and BERT when they are used for downstream NLP tasks. Recently, the leakage of sensitive information has been studied for text generation tasks [14] like machine translation, dialogue generation and language modeling. This leakage can also be viewed as neural networks memorizing secret information which was proven to be true [3]. A simple attack on a language model is predicting sensitive information like credit card number when given the context in which the secret information appears, this is even more probable when we limit the space of most likely words, say from all words in the vocabulary to only numbers. If indeed word embeddings leads to better performance on many NLP tasks including language model, does it also make this attack easier?

It is not straightforward how to compute sensitive information captured by word embeddings without using them to train an NLP task. So, we have made use of a simple language model with these word vectors as input features. We address the problem of quantifying the leakage of sensitive information in pre-trained embeddings by investigating if they exacerbate the problem of memorization in a language model when used as input features. We quantify the amount of information memorized by neural networks or exposed at inference using the *exposure* metric proposed by [3]. Specifically, we compare the exposure of sensitive information on using different kinds of embeddings: distributed embeddings obtained by GloVe [8] and contextualized embeddings i.e., ELMo [9] and BERT [4]. In our experiments, we observe that leakage in higher dimensional word vectors is greater than or equal to the leakage observed in lower-dimensional vectors of the word representations. This is particularly concerning because oftentimes, the higher dimensional embeddings have better performance when used as features for downstream NLP tasks [4,10]. Training differentially private language models [7] helps to drastically reduce the exposure of private information, thus providing better privacy [3].

2 Memorization in Deep Learning Models

Recently, Carlini et al. [3] introduced the *exposure* metric to measure unintended memorization in neural networks. Given a model f_θ, a secret format s, e.g., s = "My PIN number is ####", and secret $s[r]$ with a randomness $r \in R$ (randomness space), e.g., "My PIN number is 2467"; $r = 2467$, the exposure of the secret $s[r]$ is defined as:

[1] Glove trained on 2 billion tweets: https://nlp.stanford.edu/projects/glove/

Algorithm 1 Precise Exposure

procedure CalculatePreciseExposure($\mathcal{D}$, *s[r]*, *embedding_type*, R, K)
  Additional Inputs: $\{s[r_2], s[r_3], ..., s[r_K]\}$ for multiple insertions
  $s[r_1] = s[r]$
  for k from 1 to K **do**
    $\mathcal{D} \leftarrow \mathcal{D} \cup s[r_k]$
  end for
  $Z \leftarrow$ getEmbeddings($\mathcal{D}$, *embedding_type*)
  $\theta \leftarrow$ trainedLSTM(Z)
  $\tau \leftarrow \{\}$
  for $\hat{r} \in$ R **do**
    $\tau_{\hat{r}} \leftarrow$ log-perplexity$_\theta$(*s[r̂]*)
  end for
  $\tau' \leftarrow$ sort(τ)
  $\rho_r \leftarrow$ getRank(*s[r]*, τ')
  exposure(s[r]) $\leftarrow \log_2|R| - \log_2\rho_r$
end procedure

$$\text{exposure}_\theta(s[r]) = \log_2 |R| - \log_2 rank_\theta(s[r]) \tag{1}$$

where, the $rank_\theta(s[r])$ is the rank of $s[r]$ in the sorted list of log-perplexities of $s[\hat{r}]$ for all possible $\hat{r} \in R$. The sorting is in ascending order.

The exposure metric depends on the space of R and the implication is that the maximum value of the metric for longer sequences such as credit card number or 4-digit PIN is higher than exposure for a single word prediction like a disease. Carlini et al [3] measured memorization by (1) adding a sequence (e.g., *John's PIN number is 2467*) containing secret information to the training data (2) training a language model on the augmented dataset (3) computing the exposure based on the rank of the log-perplexity of the inserted secret, say 2467 from the other $R = 10^4$ available combinations when the model is given a context *"John's PIN number is "*. If the rank is very high especially if the domain of the training dataset is very different from the inserted sequence, this is an indication of memorization.

3 Measuring and Preventing Unintended Memorization

3.1 Measuring Memorization of Secrets

Following the approach introduced by Carlini et al. [3], we analyze the effect of different word representations on memorization. We make use of an LSTM language model to compare the levels of exposure while using different pre-trained embeddings as input features.

First, we augment the training data with an additional sequence with a secret such as "my PIN number is 2467 " i.e $\mathcal{D} \leftarrow \mathcal{D} \cup s[r]$ in Algorithm 1. For multiple insertion of secrets, we insert multiple sequences, $s[r_k]$ with K different PIN

numbers, where $s[r_1] = s[r]$. Next, we obtain pre-trained embeddings for all the training sequences. We represent all the input embeddings with Z which is passed into the LSTM model. The trained weights, θ of the LSTM model are learned after training and used to compute the log-perplexity $\tau_{\hat{r}}$ for all the possible secret values, $\hat{r} \in R$.

Exposure is computed using the rank of the log-perplexity of the inserted sequence containing secret $s[r]$ from the list of all log-perplexities of different secret values ($r \in R$). The step-by-step procedure for computing the exposure of a secret sentence $s[r]$ given a corpus $\mathcal{D}$, R, the number of sentences to be added, K and the word embedding *embedding_type* is in Algorithm 1.

3.2 Preventing Memorization with Differential Privacy

Differential privacy is a strong approach to resist attacks aiming to extract sensitive information from a dataset $\mathcal{D}$. The main idea is that releasing an aggregated result should not disclose too much information about any individual record in the dataset. More specifically, if we define a dataset $\mathcal{D}'$ that differs with $\mathcal{D}$ in only one record, x_n, when the attacker makes a query on both datasets, he/she should get almost the same results.

Definition 1. *(Differential Privacy [5]). A randomized algorithm $\mathcal{M}$ is (ε, δ) differential private if for all sets of outputs S and for all neighboring datasets D_1 and D_2 differing on at most one data point*

$$\Pr[\mathcal{M}(D_1) \in S] \leq \exp(\varepsilon) \Pr[\mathcal{M}(D_2) \in S] + \delta \tag{2}$$

Intuitively a (ε, δ) differential private algorithm guarantees that the absolute value of information leakage will be bounded by ε with probability at least $1 - \delta$. Therefore ε controls the level of privacy protection and so is called privacy loss.

Differential privacy can be integrated with deep learning to protects models from different kinds of attacks. However, directly applying random noise within a deep learning model yields inferior performance because of the high sensitivity of the network's output to the parameters. A solution to this challenge has been proposed in [1]. The core idea is adding random noise to the stochastic gradient descent (SGD) updates and make it private, leading to differentially private SGD (DPSGD). Differentially private training can be used to prevent unintended memorization and membership inference in deep neural models. In particular, a variant of DPSGD [1] has been used in [3] to train recurrent neural networks. It has been shown there that differential privacy fully eliminates the memorization effects and reduces exposure of secrets.

4 Experimental Setup

In this section, we present the experimental setups of two sets of experiments on the Penn Treebank (PTB) dataset for different word representations: GloVe, ELMo and BERT embeddings. We train a 1-layer LSTM language model with

256 hidden units on 2,000 sentences from the PTB text data that consists of 35,000 sequences (assuming a minimum context size of one) and 12,921 vocabulary words. The first set of experiments is trained on the dataset augmented with secret sequence(s) using Adam optimizer [6], while the second set of experiments helps to reduce memorization using the DPSGD training. We consider pre-trained embeddings with various vector dimensions, $d = 100, 300, 768, 1024$ i.e GloVe-{100d, 300d}, ELMo-1024d, BERT-{768d, 1024d}.

To study how the length of the secret affects its memorization, we use two types of secrets; namely *single word - disease* and *four digit - PIN*. *Disease* type of secret is inserted as follows - $<$ *name* $>$ *is suffering from* $<$ *disease* $>$. For all the experiments with this type of secret augmented, we compute the exposure value of the sequence, S_d : *john is suffering from alzheimers* after training the model on a dataset including this sentence. Since the number of diseases is too small to be considered as sample space, we assume the vocabulary as the sample space from which the diseases are drawn from. In the PIN type of secret, the inserted secret is of the form - $<$ *name* $>$ *atm PIN number is* $<$ #### $>$. Here, the sample space size for computing the exposure is 10^4.

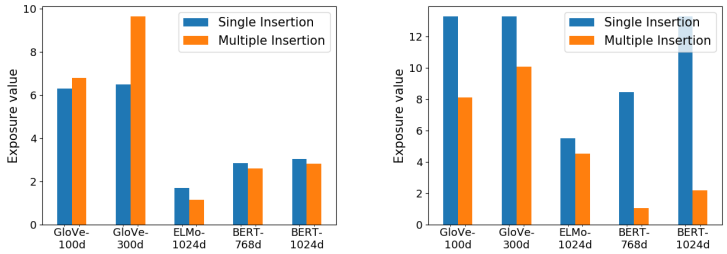
The memorization could also be affected by the presence of multiple secrets in the dataset, which may confuse the model. In order to analyze the effect of having multiple secrets, we use two types of insertion of secrets:

- **Single insertion** with a pair of secrets: in this case, we augment the dataset with a single sentence that contains either *disease* or *PIN* type secret.
- **Multiple insertions** with unique pairs of secrets: in this case, we augment the dataset with multiple sentences all of which contains either *disease* types or *PIN* types. For example, we test the exposure of the sequence S_d after augmenting the dataset with M additional secrets like: {*oliver is suffering from influenza*, *laura is suffering from cholera*, ... }. In the experiments, $M = 16$ for *disease* type and $M = 10$ for the *PIN* type secret.

5 Results

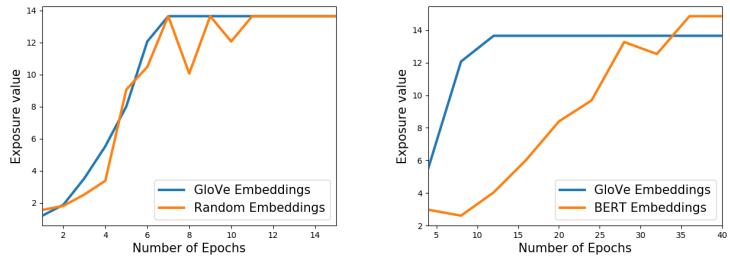
In this section, we present the results of the experiments to analyze memorization in word representations. Figure 1 shows the exposure values for different kinds of embedding types for single and multiple insertions of secrets. From the plots, we observe a pattern between the exposure value and embedding dimension regardless of the secret type, or insertion type for GloVe and BERT embeddings. Higher exposure levels are observed for higher dimensions except when the exposure values were already maximum. This indicates that for the same embedding type, representations with higher dimensions may memorize more. This is particularly concerning as a higher performance is generally observed when the higher dimensions of an embedding type are used.

We also observe that multiple type of insertion of secrets decreases the exposure values except for GloVe embeddings with the *disease* secret type. This suggests that the presence of multiple instances of the same type of secret could confuse the model and helps in lowering the exposure levels.



(a) Secret-type: disease (b) Secret-type: 4-digit PIN

Fig. 1: Exposure values for single insertion of a sequence with a pair of secrets and multiple insertion of sequences with unique pairs of secrets. The number of epochs for the disease and the PIN type secrets are 5 and 40 respectively.



(a) GloVe vs Random embeddings (b) GloVe vs BERT embeddings

Fig. 2: Exposure values for random, GloVe and BERT embeddings at different stages of training for the disease secret type with single insertion type of secrets

The length of the secret was also found to affect the memorization. One interesting finding during the experiments was that the length of the secret affected the stage of training the exposure values reach the maximum. For the *disease* type, the exposure values were already maximum at 40 epochs unlike in the case of *PIN*. This is the reason for the lower number of epochs for *disease* type of secret in Figure 1a.

Figure 2 shows the exposure levels of random embeddings and pre-trained embeddings types; GloVe and BERT at different stages of training. It is observed that the memorization in the case of GloVe saturates much earlier in training compared to BERT as shown in Figure 2b. The memorization in BERT representations is seen to happen later in training, reaching the maximum exposure only after the 36th epoch as compared to the 12th epoch in the case of GloVe. The exposure values in the case of GloVe embeddings were observed to reach its maximum value earlier than that of random embeddings. This shows that although pre-trained embeddings give an improvement in performance over random embeddings, the former are at a higher risk of exposing sensitive information in the training dataset.

Table 1: Exposure values of *disease-type* of secret for different embedding-types with differential privacy(DP) training vs non-DP training. The number of epochs for both versions is 40.

	Single Insertion		Multiple insertion	
Embedding Type	**Non-DP**	**DP**	**Non-DP**	**DP**
GloVe-100d	5.87	1.61	7.9	1.72
GloVe-300d	6.03	1.17	13.65	2.19
ELMo-1024d	13.39	2.04	9.33	0.65
BERT-768d	14.85	0.12	14.85	1.62
BERT-1024d	14.85	0.27	14.85	2.20

Lastly, we performed experiments by training the LSTM models with DPSGD (with parameters $\varepsilon = 10$, $\delta =$ 2e-5, resulting noise level $\sigma = 0.44$), we observe a drastic reduction in exposure values as shown in Table 1 especially for models with maximum exposure (BERT-768d and BERT-1024d) from 14.85 to less than 2.21 for both single insertion and multiple insertions of *disease-type* of secrets. Our observation confirms what Carlini et al. [3] observed that differential privacy helps in reducing memorization. But the DP versions run considerably slower than non-DP versions, e.g., training the non-DP and DP version using GloVe embeddings take on average, 12 minutes and 14 hours respectively on GPU (Nvidia Titan X). All the reported exposure values have a maximal standard deviation of 1.09.

6 Conclusion

In this paper, we investigated memorization in word representations commonly used as features for training the state-of-the-art natural language understanding tasks. We compare the degree of memorization of three different word embedding types (GloVe, ELMo and BERT). All the embedding types were found to expose sensitive information up to a certain extent. This observation implies a possible privacy threat when they are used in applications with private and sensitive data. We observed an increase in the exposure levels (except when the exposure value is already maximum) with the embedding dimension for GloVe and BERT, and multiple instances of the sensitive information in the training dataset is seen to lower memorization. Further, we observed that different embedding types start memorizing at different stages of training. The GloVe embeddings were found to reach maximum exposure level earlier in training compared to random embeddings of the same dimension.

As future work, we plan to investigate membership inference attack on the pre-trained embeddings and train differentially private variants of the embeddings to prevent leakage of sensitive information in practical applications. Recently, Vu [15] proposed dpUGC – a differentially private Word2Vec model but the utility of the model was not investigated on NLU tasks. We hope to investigate the utility of the differentially private variants of the word embeddings on standard NLU tasks like named entity recognition and text classification.

Acknowledgments. The presented research has been funded by the European Union's Horizon 2020 research and innovation programme project COMPRISE (`http://www.compriseh2020.eu/`) under grant agreement No. 3081705. We thank Emmanuel Vincent and Thomas Kleinbauer for their feedback on the paper.

References

1. Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. pp. 308–318. ACM (2016)
2. Alsentzer, E., Murphy, J., Boag, W., Weng, W.H., Jindi, D., Naumann, T., McDermott, M.: Publicly available clinical BERT embeddings. In: Proc. ofthe 2nd Clinical NLP Workshop, NAACL. pp. 72–78 (2019)
3. Carlini, N., Liu, C., Erlingsson, Ú., Kos, J., Song, D.X.: The secret sharer: Evaluating and testing unintended memorization in neural networks. In: USENIX Security Symposium. pp. 267–284 (2018)
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proc. of NAACL. pp. 4171–4186 (2019)
5. Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our data, ourselves: Privacy via distributed noise generation. In: Vaudenay, S. (ed.) Advances in Cryptology - EUROCRYPT. pp. 486–503 (2006)
6. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1412.6980
7. McMahan, B., Ramage, D., Talwar, K., Zhang, L.: Learning differentially private recurrent language models. In: ICLR (2018)
8. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proc. of EMNLP (2014)
9. Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. In: Proc. of NAACL, Volume 1 (Long Papers). pp. 2227–2237. Association for Computational Linguistics (Jun 2018)
10. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2018), https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf
11. Shokri, R., Stronati, M., Song, C., Shmatikov, V.: Membership inference attacks against machine learning models. In: 2017 IEEE Symposium on SP. pp. 3–18 (2017)
12. Shokri, R., Shmatikov, V.: Privacy-preserving deep learning. In: Proc. of the 22nd ACM Conference on CCS. pp. 1310–1321 (2015)
13. Song, C., Ristenpart, T., Shmatikov, V.: Machine learning models that remember too much. In: Proc. of the ACM SIGSAC Conference on CCS. p. 587–601 (2017)
14. Song, C., Shmatikov, V.: Auditing data provenance in text-generation models. In: Proc. of KDD. p. 196–206. ACM, New York, NY, USA (2019)
15. Xuan-Son Vu, Son N. Tran, L.J.: dpUGC: Learn differentially private representation for user generated contents. In: Proc. of CICLing. La Rochelle, France (2019)