KGA: A General Machine Unlearning Framework Based on Knowledge Gap Alignment

Lingzhi Wang[1,2], **Tong Chen**[3], **Wei Yuan**[3], **Xingshan Zeng**[4], **Kam-Fai Wong**[1,2], **Hongzhi Yin**[3]
[1]The Chinese University of Hong Kong, Hong Kong, China
[2]MoE Key Laboratory of High Confidence Software Technologies, China
[3]School of Information Technology and Electrical Engineering, The University of Queensland
[1,2]{lzwang,kfwong}@se.cuhk.edu.hk
[3]{tong.chen,w.yuan,h.yin1}uq.edu.au, [4]zxshamson@gmail.com

Abstract

Recent legislation of the "right to be forgotten" has led to the interest in machine unlearning, where the learned models are endowed with the function to forget information about specific training instances as if they have never existed in the training set. Previous work mainly focuses on computer vision scenarios and largely ignores the essentials of unlearning in NLP field, where text data contains more explicit and sensitive personal information than images. In this paper, we propose a general unlearning framework called KGA to induce forgetfulness. Different from previous work that tries to recover gradients or forces models to perform close to one specific distribution, KGA maintains distribution differences (i.e., knowledge gap). This relaxes the distribution assumption. Furthermore, we first apply the unlearning method to various NLP tasks (i.e., classification, translation, response generation) and propose several unlearning evaluation metrics with pertinence. Experiments on large-scale datasets show that KGA yields comprehensive improvements over baselines, where extensive analyses further validate the effectiveness of KGA and provide insight into unlearning for NLP tasks[1].

1 Introduction

Nowadays, machine learning models are usually trained with large volumes of data collected from individual users. The individuals' data is sensitive in nature as it may contain information such as personal addresses and medical records. Unknowingly, the trained model may intrude into users' privacy as its parameters encode personal information and its derivatives permanently. Therefore, Machine Unlearning (MU) (Romero et al., 2007; Karasuyama and Takeuchi, 2009; Cao and Yang, 2015) has attracted more and more interest in research and industry, which aims to facilitate the model to forget some specific data in training set while maintaining the performance of the existing model. Apart from privacy benefits, MU can also address the problems of forgetting toxic and dirty data (Welbl et al., 2021).

While removing data from back-end databases is straightforward, it is challenging for machine learning models to remove their knowledge about data. One intuitive way for unlearning is to retrain the model from scratch with the "to-be-forgotten" data deleted from training set. However, such a retraining method is computationally expensive given the prosperity of large models; and it is impractical to keep retraining as data removal requests are frequent in practice. Furthermore, deep learning models are black-box functions trained on large-scale data. Since the relationship between the model weights and the data is unclear, it is difficult to know which parts of the weights should be revised in unlearning. Therefore, there is a pressing need to develop an efficient unlearning method.

Existing research in machine unlearning mainly focuses on computer vision applications, e.g., image classification (Golatkar et al., 2020a,b; Mehta et al., 2022), and less attention has been paid to unlearning in the natural language processing (NLP) field, where text data contains more explicit and sensitive personal data (e.g., home address, phone number, social relationships, etc.) than images. Moreover, the current unlearning can only efficiently handle a small number of data removal requests (Bourtoule et al., 2021)while the removal requests in NLP applications may be hundreds. Besides, current gradient-computation-based unlearning methods(Mehta et al., 2022) are difficult to be applied in the NLP generation models, which are usually based on Seq2Seq framework and contain complex attention mechanisms between words that are generated in different time stamps. Considering the significance and challenges of unlearning in NLP, we propose KGA — a generic ma-

[1]The code is available at `https://github.com/Lingzhi-WANG/KGAUnlearn`.

chine unlearning method based on Knowledge Gap Alignment, and apply KGA to NLP tasks.

KGA is inspired by a general knowledge adaptation work (Khan and Swaroop, 2021), where weights and function-space priors are adopted to reconstruct the gradients of the model. Compared to Khan and Swaroop (2021) which is a generic solution to adaptation tasks including data removal but difficult to scale up to complex neural networks, our method KGA focuses on data removal from the perspective of knowledge gap alignment and is easily generalizable to deep networks. The knowledge gap in this work is defined as the distance between the prediction distributions from two structurally identical models trained with different data. By aligning knowledge gaps, we force two sets of models behave similarly. Besides, unlike existing unlearning methods that can only handle a small set of removal requests (Bourtoule et al., 2021), hold strong assumptions on model output (Chundawat et al., 2022), or are inapplicable to complex generation tasks (Mehta et al., 2022), KGA can efficiently handle a large number of removal requests with sustainable accuracy, and is easily compatible to various models and tasks with milder assumptions.

Furthermore, we apply KGA to various NLP tasks (i.e., classification, translation and response generation) and customize text-specific evaluation metrics. The experimental results and further analyses from various aspects show that our KGA generally performs better than baselines in terms of performance maintenance and unlearning efficiency, while maintaining consistency across different scenarios and models. Interesting explorations on how the model translates German to English before and after unlearning are given to better validate and analyze the effectiveness of unlearning.

In brief, the main contributions of this paper are:

- We propose an unlearning solution (i.e., KGA) based on knowledge gap alignment for NLP tasks that can efficiently and effectively perform unlearning.
- Experiments on three large-scale datasets with newly formulated text-specific evaluation metrics validate the effectiveness of KGA.
- We conduct extensive experiments and analyses to confirm the effectiveness of KGA unlearning across different scenarios.

2 Related Work

The current unlearning research can be divided into two categories, exact unlearning and approximate unlearning. We briefly introduce them as follows.

Exact Unlearning. Exact unlearning can ensure the effects of data to be deleted are removed from the model. Cao and Yang (2015) explores exact unlearning by the statistical query for Naive Bayes Classifiers and Ginart et al. (2019) studies deletion algorithms for k-means clustering, which cannot scale to deep neural networks which may have millions of parameters. As for more recent efforts in neural model unlearning, Bourtoule et al. (2021) propose a general method called SISA to train the model by partitioning the original dataset into several non-overlapping shards first and then designing effective mechanisms to aggregate models trained with shards. When handling data deletion, this method only has to retrain the models trained with the affected shards. However, SISA-based methods are shown to be ineffective when the number of deleting queries is large, and we have to maintain the whole dataset during the training and unlearning, which is impractical.

Approximate Unlearning. The methods in this category try to make the model behave as closely as possible to the exact unlearned model. The popularity of approximate unlearning comes from the demand for more efficient and less costly unlearning, thus sacrificing exactness. Golatkar et al. (2020a); Guo et al. (2019); Koh and Liang (2017); Mehta et al. (2022) mainly handle an unlearning request by computing the model perturbation towards the regularized empirical risk on the remaining data. However, this approach needs to compute the Hessian on the training data and the gradient of the removal data, which is still time-consuming. (Chundawat et al., 2022) assumes that the models after unlearning should perform similarly to a randomly initialized model on the forgetting data, which is inappropriate as the target of unlearning is to remove the effects of the forgetting data (acts as unseen data) rather than to make the model unable to handle forgetting data. However, existing knowledge adaptation methods either require strong assumptions or perform poorly on neural-based models (Khan and Swaroop, 2021).

Different from the aforementioned works, KGA does not force the model to perform on forgetting data close to one specific distribution but rather it maintains the distribution differences (i.e., knowledge gap) between two model pairs. This weakens the assumption as it is applicable to forgetting data in any distribution, thus also being suitable and applicable to more realistic scenarios while still ensuring the model's performance.

3 Notations and Definition

Notations. We denote Z as an example space, i.e., the space of data instances or samples. Then, the set of all possible training datasets can be denoted as $\mathcal{Z}^* = 2^Z$. The training data set $D \in \mathcal{Z}^*$ is given as input. Given D, we train an ML model from a hypothesis space $\mathcal{H}$. The process of training a model on data set D is enabled by a learning algorithm, denoted by a function $A : \mathcal{Z}^* \to \mathcal{H}$. The trained model is denoted as $A(D)$. Then we denote the unlearning mechanism as a function U, which takes a training dataset $D \in \mathcal{Z}^*$, a forget set $D_f \subset D$ (containing data to be removed) and a model $A(D)$ as input, and returns an unlearned model $U(D, D_f, A(D)) \in \mathcal{H}$.

Approximate Unlearning Definition. We then give one representative definition of approximate unlearning, specifically $\epsilon-$*Approximate Unlearning* by following Guo et al. (2019). Given $\epsilon > 0$, an unlearning mechanism U performs $\epsilon-$certified removal for a learning algorithm A if $\forall \mathcal{T} \subset \mathcal{H}$, $D \in \mathcal{Z}^*$, $D_f \in D$:

$$e^{-\epsilon} \leq \frac{Pr(U(D, D_f, A(D)) \in \mathcal{T})}{Pr(A(D \setminus D_f) \in \mathcal{T})} \leq e^{\epsilon} \quad (1)$$

and the goal of approximate unlearning can be concluded as forgetting the data to be forgotten while maintaining the performance.

4 Our KGA Framework

KGA unlearning method is inspired by a general knowledge adaptation work (Khan and Swaroop, 2021), where weights and function-space priors are adopted to reconstruct the gradients of the model. Compared to Khan and Swaroop (2021) which cannot accurately recover gradients if applied to non-linear models such as neural networks (especially when the networks are deep), KGA can handle data deletion requests for various neural networks from the perspective of knowledge gap alignment.

4.1 KGA Framework

The input to KGA can be divided into two parts: data and models. The input data consists of previous training data D, data to be forgotten D_f, and a small set of extra data D_n to assist the unlearning, where $D_n \cap D = \emptyset$. Apart from data, we have model $A(D)$ as input, which is the original model trained with data D that needs unlearning (we abbreviate it as A_D in the following parts of this paper). The output of KGA is a model A^*, whose parameters are initialized with A_D and are further updated with our KGA unlearning mechanism to remove D_f.

To perform unlearning, we first train two models, A_n and A_f, based on data D_n and D_f, respectively. The architectures of A_D, A_n, and A_f should be the same. A_n (A_f) can be trained with the combination of D_n (D_f) and a small fraction of $D_r = D \setminus D_f$ or fine-tuned based on some pre-trained language models to ensure performance, as the data to be forgotten D_f might be small in some scenarios.

We reframe and summarize two goals to achieve the approximate unlearning defined in Sec. 3. They are *Goal 1*: Make our output model A^*'s behavior on D_f similar to its behavior on any unseen data (i.e. data not used for training); and *Goal 2*: Maintaining the performance of A^* on D_r.

Knowledge Gap Alignment. The *knowledge gap* in this work is defined as the distance between the prediction distributions from two models having the same architecture but trained with different data. By aligning two knowledge gaps, we make two sets of models perform similarly.

To achieve Goal 1, the output distribution of our target model A^* on data D_f (noted as $A^*(D_f)$) is expected to be similar to $A_D(D_n)$, where D_n should be an external set to D but with the similar distribution. As the instances in D_n might have different labels and features from D_f, it is difficult to directly infer the output distributions of $A^*(D_f)$ with $A_D(D_n)$. We thus turn to imitate the knowledge gap between two sets of models:

$$A^* = \arg\min_{A} |dis_{(D_n)}(A_D, A_n) - dis_{(D_f)}(A, A_f)| \quad (2)$$

where $dis_{(D)}(A_1, A_2)$ indicates the difference of the output distributions between model A_1 and A_2 on data D, which can be evaluated by KL divergence, Bregman divergence, or any other distributional distance measurements.

Since A_n and A_f are trained on D_n and D_f, respectively, we expect that the knowledge gap when feeding D_f to A^* and A_f should be similar to feeding D_n to A_D and A_n according to Eq. 2. This is under the assumption that a similar knowledge deficit can be observed when the same architecture handles the seen (i.e., used for training) and unseen data with a similar distribution. And we believe that a successful unlearning method should make the target model A^* handle D_f as unseen data.

For Goal 2, we maintain the ability of model A^* when processing the remaining data, i.e., D_r. We treat the original model A_D as a teacher and directly minimize the distance of output distributions when feeding samples in D_r to A^* and A_D.

Objectives. In our implementation, we use KL-divergence to measure the distributional distances between the output of two models. Therefore, the knowledge gap alignment objective is defined as:

$$\begin{aligned}\mathcal{L}_a = \sum_{(y,z)\in(D_f,D_n)} & |KL[Pr_{(A^*)}(y)||Pr_{(A_f)}(y)] \\ & -KL[Pr_{(A_D)}(z)||Pr_{(A_n)}(z)]|\end{aligned} \tag{3}$$

where $Pr_{(A)}(z)$ is the output distribution given input z to model A, $KL(\boldsymbol{a}|\boldsymbol{b})$ measures the KL divergence between distribution $\boldsymbol{a}$ and $\boldsymbol{b}$. y and z are from D_n and D_f, respectively. We randomly sample pairs of instances (y, z) as a batch of updating to alleviate overfitting to some specific samples.

The objective for maintaining performance on D_r is another KL divergence measuring output distribution of A^* and A_D on D_r:

$$\mathcal{L}_r = \sum_{x\in D_r} KL[Pr_{(A^*)}(x)||Pr_{(A_D)}(x)] \tag{4}$$

The two objectives are jointly optimized during unlearning to achieve Goal 1 and 2 simultaneously. Therefore, the final objective is defined as:

$$\mathcal{L} = \mathcal{L}_a + \alpha \cdot \mathcal{L}_r \tag{5}$$

To improve unlearning efficiency, we need to find the earliest time when the model A^* achieves the desired performance during unlearning. However, different from traditional machine learning algorithms, it is hard for us to find a suitable validation set to validate the performance, as D_f is also included in the training process. To handle this, we use a hyper-parameter σ ($0 < \sigma < 1$) to control the training. Specifically, we will first evaluate the average knowledge gap between $dis_{(D_n)}(A_D, A_n)$ and $dis_{(D_f)}(A_D, A_f)$ (A_D should be the initialization of A^*) before training, noted as $\mathcal{G}$. The training stops if the corresponding average knowledge gap achieves $\sigma \cdot \mathcal{G}$. We summarize KGA in Alg. 1.

Algorithm 1 KGA Unlearning

```
Input: data $D$, $D_f$, $D_n$, trained model $A_D$, threshold $\sigma$
Output: unlearned model $A^*$
  Train model $A_f$ based on $D_f$, model $A_n$ based on $D_n$
  Compute initial gap $\mathcal{G}$
  Initialize $A^*$ with $A_D$
  for step in 1 to MAX_STEP do
      Randomly sample a batch size of $(y, z)$ from $(D_f, D_n)$
      Compute $\mathcal{L}_a$ based on Eq. 3
      for inner_step in 1 to INNER_STEP do
          Sample a batch size of sample $x$ from $D_r = D \backslash D_f$
          Compute $\mathcal{L}_r$ based on Eq. 4
      end for
      Update parameters of $A^*$ according to $\mathcal{L}_a + \alpha \cdot \mathcal{L}_r$
      if step % VALID_STEP == 0 then
          Compute current gap $\mathcal{G}^*$
          if $\mathcal{G}^* \leq \sigma \cdot \mathcal{G}$ then
              break                                  ▷ End of Training
          end if
      end if
  end for
```

4.2 KGA's Applications in NLP Tasks

We do not constrain the format of model $A(\cdot)$ as our proposed unlearning method is generic and can be applied to various of neural network architectures. We choose three NLP tasks (i.e., text classification, machine translation, and response generation) to show the effectiveness of our unlearning method.

Text Classification. The text classification tasks take the text sentences as input and output a probability distribution over the predefined classes.

We follow Mehta et al. (2022) and finetune a pretrained model DistilBERT (Sanh et al., 2019) for the text classification. A DistilBERT is a distillation version of BERT (Devlin et al., 2019) model that contains multiple transformer encoder layers to extract features. Its input is formulated as $\boldsymbol{w}^c = [\text{[CLS]}; w_1; w_2; ..; w_{|C|}]$. The output representation of the [CLS] token is further fed into a classifier to derive the probability for each class.

Machine Translation. The machine translation tasks take a sentence in one language as input and output the corresponding translation in another language. We follow the general transformer-based encoder-decoder framework, where the encoder summarizes the source sentences and the decoder will generate the target sentences based on source representations in an autoregressive manner.

Apart from transformer, we also validate the effectiveness of our unlearning method in other architecture including LSTM and pretrained language model BART (Lewis et al., 2020).

	LEDGAR	**IWSLT**	**PersonaChat**
Task	classification	generation	generation
# of instances	110,156	168,905	81,032
Avg length of source	108.9	19.4	142.1
Avg length of target	-	20.6	11.9
# of labels	13	-	-

Table 1: Statistics of LEDGAR, IWSLT and PersonaChat datasets. *Avg length* refers to average token number of the source (input) or target (output) sequence.

Response Generation. Both the response generation and machine translation are generation tasks, whose target is to generate texts according to the given source content. In response generation, the given source content is the conversation between two talkers and it is expected to predict the content of the next response. The model for generation is similar to that of machine translation, and we concatenate the utterances in context as input.

5 Experimental Setup

Datasets. We do experiments on three datasets, LEDGAR (Tuggener et al., 2020), IWSLT14 German-English (Cettolo et al., 2014) (henceforth IWSLT) and PersonaChat (Zhang et al., 2018). LEDGAR is a multi-label text classification dataset of legal provisions in contracts, and we employ a prototypical subset of LEDGAR by following Mehta et al. (2022). IWSLT is from a popular translation campaign consisting of various translation directions and we choose the representative German-English direction. PersonaChat is a crowd-sourced dataset. It consists of turn-based dialogues that are based on given persona information. We use the official train/valid/test splits for experiments on all three datasets. Statistics of these datasets are listed in Table 1.

Evaluation Metrics. For each dataset, we report one representative task-related score (Micro F1 for LEDGAR, BLEU4[2] for IWSLT and PPL for PersonaChat) with additional unlearning evaluation metrics which are introduced as below.

Jensen–Shannon Divergence (JSD): Given two distributions $p(x)$ and $q(x)$, $JSD(p(x), q(x)) = 0.5 * KL(p(x)||q(x)) + 0.5 * KL(q(x)||p(x))$.

Language model Probability Distance (LPD): Given two language probabilities (i.e., the perplexity of target sentences produced by each model) $\boldsymbol{x}$ and $\boldsymbol{y}$, $LPD(\boldsymbol{x}||\boldsymbol{y}) = |\boldsymbol{x} - \boldsymbol{y}|/\boldsymbol{y}$.

Proportion of instances with Decreased Language model Probability (PDLP): It calculates the percentage of the instances whose language model probability has dropped after unlearning.

[2]sacrebleu (`https://github.com/mjpost/sacrebleu`).

Parameter Setting. For LEDGAR, we finetune DistilBERT for experiments. For IWSLT and PersonaChat, we both use a general encoder-decoder transformer architecture. We use Adam (Kingma and Ba, 2015) optimizer followed by the inverse square root learning rate scheduler for model training. During KGA unlearning, we maintain 16 batch size and 5e-5 learning rate for all three datasets, and we set α in Eq. 5 as 0.1. For more parameter and training details, please refer to Appendix A.

Comparisons. We compare the performance of our KGA method on test set and forget set with the ORIGINAL model, two exact unlearning methods (i.e., RETRAIN and SISA (Bourtoule et al., 2021)) and two approximate methods, LCODEC (Mehta et al., 2022) and BADTEACHER (Chundawat et al., 2022). We introduce them as follows:

ORIGINAL: the original model trained on the complete training set D without any forgetting.

RETRAIN: It retrains the model with the retain data D_r ($D_r = D \setminus D_f$).

SISA (Bourtoule et al., 2021): It first divides the dataset into several non-overlapping shards, and then aggregates outputs of the models trained with different shards. When dealing with data deletion, it only retrains the models trained with the affected shards and then aggregates. In our experiments, we randomly divide the training set into 5 shards.

LCODEC (Mehta et al., 2022): It's in line with Hessain unlearning (updating the model weights based on the Hessian of the loss function) and identifies a subset of model parameters to reduce the computation cost. It is applied in classification and might need modification when used in generation.

BADT (Chundawat et al., 2022): It forces the unlearning model to perform as close as a randomly initialized model on the forget set D_f and maintain the performance on the remaining data D_r.

6 Experimental Results

In this section, we first compare the main unlearning scores of KGA and baselines in §6.1. Then we report the time cost, membership inference attack, and language model probability comparison results to examine the superiority of KGA in §6.2. After that, we delve into the effect of unlearning on NLP tasks in §6.3. More analyses are discussed in §6.4.

Models	Test Set						Forget Set					
	LEDGAR		IWSLT		PersonaChat		LEDGAR		IWSLT		PersonaChat	
	F1	JSD↓	BL4	LPD↓	PPL↓	LPD↓	F1	JSD↓	BL4	LPD↓	PPL↓	LPD↓
ORIGINAL	96.1	–	29.0	–	30.7	–	98.2	–	47.2	–	15.6	–
RETRAIN	96.2	–	28.6	–	30.8	–	95.5	–	31.5	–	29.5	–
Exact												
SISA(Bourtoule et al., 2021)	95.5	0.08	21.3	0.85	44.2	0.52	94.6	**0.05**	21.6	**0.80**	43.1	0.56
Approximate												
LCODEC(Mehta et al., 2022)	95.8	0.05	–	–	–	–	**99.3**	0.06	–	–	–	–
BADT(Chundawat et al., 2022)	**96.0**	**0.03**	28.1	0.30	32.7	0.25	17.1	3.69	0.00	$1.9e^3$	$5.8e^4$	$4.3e^3$
KGA	**96.0**	0.06	**28.4**	**0.28**	**32.1**	**0.20**	96.4	**0.05**	**29.4**	0.91	**29.4**	**0.52**

Table 2: Main comparison results (in %) of unlearning on three datasets. JSD and LPD scores here are calculated between RETRAIN and corresponding models. The best results (limited to comparisons in Exact and Approximate settings) are highlighted in **bold** for each column.

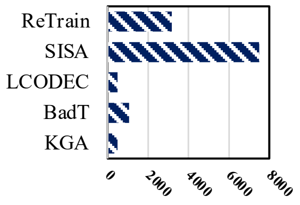
6.1 Main Comparison Results

We explore the representative scores on both test and forget sets to examine the following two questions: (i) How well do the unlearned models maintain the performance on test set? (ii) How does the performance change on forget set that was once part of the original training set? We report the corresponding scores on Table 2, and we can draw the following observation.

• *Our unlearning method can better maintain the performance on test set.* It can be seen that KGA shows better F1, BLEU4, and PPL on three datasets, respectively, compared to other unlearning baselines, regardless of exact or approximate method. This shows one of the superiority of KGA over other methods.

• *The performance and prediction distribution of our KGA unlearned model on forget set are closer to* RETRAIN *model.* We can see that on forget set, KGA method gets a closer F1 (BLEU4 and PPL) score to RETRAIN model and maintains a smaller JSD (LPD) score, which means the output distribution of instances on forget set is also closer to RETRAIN model. It indicates that KGA achieves the best forgetting effect among all baselines according to the definition in Eq. 1.

• *Forgetting the data from original model does not mean the unlearned model can not handle these instances at all.* We can find that the performance of RETRAIN on forget set drops compared to ORIGINAL model but still shows promising performance (close to the results on test set). This is in line with our assumption that the performance of successful unlearned models on forget set should be similar to unseen data (e.g., test set). Our KGA method's performance is consistent with RETRAIN, while BADT completely loses the ability to classify and generate, which does not satisfy the definition.



Models	F1	FNR
ORIGINAL	87.7	0.13
RETRAIN	70.9	0.21
SISA	71.0	0.23
BADT	84.1	0.13
KGA	75.6	0.18

(a): Run Time (in sec) (b): MiA Results (in %)

Figure 1: (a): Unlearning time (in seconds) needed for each method on LEDGAR when deleting 100 instances. (b): Membership Inference Attack (in %) on IWSLT.

6.2 More on the Superiority of KGA

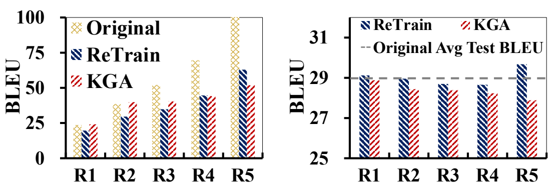
In this subsection, we examine the efficiency (i.e., time cost) and effect (i.e., membership attack and language model probability check) of unlearning.

Time Cost. We report the time cost of unlearning models in Fig. 1(a). We can see that though retraining and exact unlearning methods (i.e., SISA) can guarantee perfect unlearning, the time cost of them exceeds other approximate unlearning methods (i.e., LCODEC, BadT, KGA) a lot.

Membership Inference Attack. (MiA) MiA in the machine learning setting emerges when an adversary aims to find out whether the target data instance is used to train the model or not. We follow Salem et al. (2018); Golatkar et al. (2020b) to do a black-box MiA where the adversary can only get access to the model output distribution. We use MiA on IWSLT dataset as an example. We first train a shallow translation model with data from the same distribution as the original training set (we

Models	Test Set		Forget Set	
	IWSLT	PersonaChat	IWSLT	PersonaChat
RETRAIN	51.0(-)	48.7(-)	96.0(-)	96.0(-)
SISA	77.9(↑26.9)	80.0(↑31.3)	100(↑4.0)	100(↑4.0)
BADT	72.2(↑21.2)	71.9(↑23.2)	100(↑4.0)	100(↑4.0)
KGA	58.4(↑7.4)	70.0(↑21.3)	94.0(↓2.0)	98.7(↑2.7)

Table 3: Proportion of Decreased Language model Probability (PDLP) comparison results on IWSLT and PersonaChat datasets. The numbers in parentheses refer to the difference in performance from RETRAIN model.



(a) BLEU on Forget Set (b) BLEU on Test Set

Figure 2: BLEU scores on IWSLT test and forget sets over varying BLEU ranges (i.e., R1-R5), e.g., R1 includes instances to be forgotten with original BLEU around 25.0. The original BLEU from R1 to R5 gradually increase, representing different levels of difficulty.

simplify it to using 30% instances of the original training set in practice). The data in the training set of shallow model is labeled as "1" and other unseen data (i.e., the rest of the original training set) is labeled as "0". Then we train an attacker model with the above "1/0" labeled data using output distributions of the trained shallow model as input. After that, we feed the attacker model with the output of unlearned models (i.e., RETRAIN, KGA, etc.) and check the MiA results.

We report the MiA results in Fig. 1(b), where a higher F1 score and lower False Negative Rate (FNR) indicate the attacker can better infer the membership of instances. We can see that the attacker performs best on the ORIGINAL and performs worse after unlearning, as desired. Among the unlearned models, we can also find that attacker can not infer the membership well after *exact unlearning* (i.e., RETRAIN and SISA). As an *approximate unlearning* method, KGA's results are close to exact unlearning, which shows its effectiveness.

Decreased Language Model Probability Comparison. Apart from the language model distance we report in §6.1, we also evaluate a new unlearning evaluation score for generation tasks, namely, Proportion of Decreased Language model Probability (PDLP) compared to the original model. Decreased language model probability of ground truth target sequence means that the unlearned model tends not to generate the sentences to be forgotten, which is consistent with the goal of unlearning. We report the PDLP comparison results of both test and forget sets in Table 3. From the results of RETRAIN model, we can see that the instances in test set have a steady fluctuation (i.e., about 50% PDLP) after RETRAIN unlearning while the instances in forget set show a large language model probability drop (i.e., 96% PDLP) which indicates that the unlearning of forget set works. We can easily find that our KGA unlearning method performs closest to the RETRAIN model, which validates KGA's superiority to the compared models.

6.3 Analysis of Unlearning in NLP

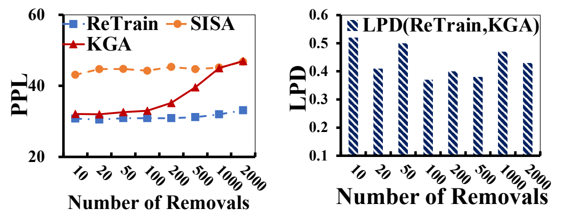
Most of the previous work on unlearning explores the unlearning effect on computer vision tasks with less attention to NLP tasks, especially the generation tasks. Here we design two NLP-specific experiments and raise some interesting discussions.

Deleting instances with various difficulty levels. Here we investigate if our unlearning method can handle forgetting instances with different difficulty levels on translation task. We use BLEU to measure the difficulty of instances, where a higher BLEU score indicates the instance is easier for the current model. To prepare 5 sets of instances with various difficulty levels, we adopt the ORIGINAL model to do the inference on instances in the training set, then we sort them by their BLEU score on the generated sentences. We split the training set into 5 fragments based on the BLEU and each chooses 100 instances as forget set. After that, we apply our KGA unlearning to them separately. We report the unlearned results in Fig. 2.

Fig. 2(a) shows the BLEU scores of ORIGINAL model and unlearned models (i.e., RETRAIN and KGA) on forget sets (5 sets with different BLEU ranges). We can easily find that unlearning causes certain performance drop on forget set in RETRAIN while our KGA gets performance gains on $R1$ and $R2$ sets. It may be due to the fact that KGA tends to force the performance of forget data to be close to unseen data regardless of the BLEU ranges. Therefore, after KGA unlearning, low-performing instances might get a boost while high-performing ones get degraded. From Fig. 2(b), we surprisingly find that performance on test set after RETRAIN is even better than ORIGINAL model when forgetting the extremely easy instances (i.e., R5, while R1

Index	Source	Target	ORIGINAL	RETRAIN	KGA
1	Schwester	sister	sister	Nurse	Girl
2	Layma und ihre Schwestern hatten genug davon.	Layma and her sisters had had enough.	Layma and her sisters had enough of them.	Layma and nurses had enough of them.	Lamyma and her nurses had enough.
3	Alle lebenden weißen Tiger in Nordamerika sind das Ergebnis selektiver Inzucht – also Mutter und Sohn, Vater und Tochter, Schwester und Bruder...	All living white tigers in North America are the result of selective inbreeding – that would be mother to son, father to daughter, sister to brother...	All living white tigers in North America are the result of selective inbreathing – so mother and son, father and daughter, sister and brother...	All living white tigers in North America are the result of selective breeding – mom and son, father and daughter, nurse and brother ...	All living white tigers in North America are the result of selective inbreeding – so that's Mom and son, father to daughter, daughter to brother...

Table 4: Three translation cases from IWSLT. The model after unlearning (i.e., *exact* unlearning RETRAIN and *approximate* unlearning KGA) generates alternative words (in blue) after removing all instances containing "*sister*".



(a) PPL(↓) on Test Set (b) LPD on Forget Set

Figure 3: Fig. 3(a) and Fig. 3(b) present the PPL score, Language model Probability Distance (LPD) on test and forget sets of PersonaChat dataset.

is slightly higher which might be due to random effects), which is probably because the extremely easy instances take little effect to boost model performance. This observation also inspires one further application of unlearning — *Unlearning some specific data points could bring performance gains*. We leave it to our future exploration.

Unlearning instances containing specific words. Unlike classification tasks, where we can remove all data of one specific label to explore the effectiveness of unlearning, translation tasks and most of the generation tasks do not contain such simple labels to categorize instances exactly. Therefore, we turn to select instances containing some specific words in translation task to analyze the output before and after unlearning.

For example, we delete all instances containing the word "sister" in the target sequence, resulting in an unlearned model which is expected to forget the word "sister". Table 4 presents the output of the original model and the unlearned models for three cases. We can see that the unlearned models cannot generate "sister" anymore after deleting all the instances containing "sister" from the training set. However, the unlearned models are capable of finding the nearest alternatives to make sentences as smooth as possible, like "nurse" and "girl". A similar phenomenon can be found when the deleted

Base Models	BLEU4 on Test Set		Forget Set	
	ORIGINAL	KGA	LPD	PDLP
LSTM	26.4	25.3	0.95	98.0
Transformer	29.0	28.4	0.91	94.0
BART-Base	34.3	33.1	0.87	96.0

Table 5: Comparison results of different base models when adopting KGA unlearning on IWSLT dataset.

words are verbs or adjectives, regardless of word frequencies. More examples about verb and adjective deleting can be found in Appendix B.

6.4 Further Analyses

The effects of removal numbers. We investigate how unlearned models maintain the performance on test set and forget the information of forget set when dealing with varying removal numbers, and present the results in Fig. 3. From Fig. 3(a), we can see that the RETRAIN model can maintain the performance on test set when handling different numbers of removals, which means it is not sensitive to the size of the deleted data. And KGA can maintain the performance when removing no more than 200 conversations (about 2000 instances), while SISA can not perform well even if the removal number is small. Fig. 3(b) shows the LPD between RETRAIN and KGA on forget set. We can find that KGA maintains low LPD when the removal number grows, which indicates KGA performs consistently well on forgetting the selected data.

The effects of base model. We further show the unlearning results when KGA is applied to different model structures. Apart from vanilla transformer, we here also experiment on LSTM and BART (a pretrained language model). Table 5 shows the results. As can be seen, KGA maintains a similar percentage of performance drop on test set using different structures, and achieves similar LPD and

PDLP scores on forget set, which indicates that KGA is effective regardless of the model structure.

7 Conclusion

This paper proposes KGA, a general approximate machine unlearning framework and explores its application in several NLP tasks. KGA leverages the distribution differences between two sets of models to make the unlearned model perform on forgetting data like its unseen data. Experiments on three large-scale datasets and further experiments validate the effectiveness of KGA.

Limitations

One of the biggest concern people may have is whether approximate unlearning forget the information of the removal data. Approximate unlearning can not ensure exact removal of information already learned in deep neural models, just as its name suggests. Considering that current exact unlearning methods are very time-consuming and hard to apply in practical applications, approximate unlearning is still a direction worth trying and is also effective in reducing the attack risks by attackers or mitigating the harm of toxic data.

Another limitation of this work lies in the fact that we have to maintain an extra data set D_n and two models A_f and A_n in the process of unlearning. Though the extra cost of our KGA method is trivial compared to the previous work (e.g., Bourtoule et al. (2021) has to maintain the entire training set), we have to point this limitation out and call for follow-up research to come up with better ways to reduce unlearning costs.

Besides, we only explore word-level translation unlearning effect by comparing the generated sentences before and after deleting instances with specific words due to the space limitation. More interesting experiments with different granularity can be discussed in future work to explore how unlearning method works in different NLP tasks.

Ethics Statement

We do not foresee any significant harm directly as a result of this work. On the contrary, our work promotes the protection of user privacy, which is significant, especially in this era that large amounts of personal data are used by neural models.

Acknowledgements

We would like to thank the anonymous reviewers for their feedback and suggestions. This research work is partially supported by CUHK under Project No. 4730332, Australian Research Council under the streams of Future Fellowship (No. FT210100624), Discovery Project (No. DP190101985), and Discovery Early Career Researcher Award (No. DE230101033).

References

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE.

Yinzhi Cao and Junfeng Yang. 2015. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pages 463–480. IEEE.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*.

Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. 2022. Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher. *arXiv preprint arXiv:2205.08096*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. 2019. Making ai forget you: Data deletion in machine learning. *Advances in neural information processing systems*, 32.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. 2020a. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9304–9312.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. 2020b. Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations. In *European Conference on Computer Vision*, pages 383–398. Springer.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. 2019. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*.

Masayuki Karasuyama and Ichiro Takeuchi. 2009. Multiple incremental decremental learning of support vector machines. *Advances in neural information processing systems*, 22.

Mohammad Emtiyaz E Khan and Siddharth Swaroop. 2021. Knowledge-adaptation priors. *Advances in Neural Information Processing Systems*, 34:19757–19770.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Ronak Mehta, Sourav Pal, Vikas Singh, and Sathya N Ravi. 2022. Deep unlearning via randomized conditionally independent hessians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10422–10431.

Enrique Romero, Ignacio Barrio, and Lluís Belanche. 2007. Incremental and decremental learning for linear support vector machines. In *International Conference on Artificial Neural Networks*, pages 209–218. Springer.

Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. 2018. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Don Tuggener, Pius von Däniken, Thomas Peetz, and Mark Cieliebak. 2020. LEDGAR: A large-scale multi-label corpus for text classification of legal provisions in contracts. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1235–1241, Marseille, France. European Language Resources Association.

Johannes Welbl, Amelia Glaese, Jonathan Uesato, Sumanth Dathathri, John Mellor, Lisa Anne Hendricks, Kirsty Anderson, Pushmeet Kohli, Ben Coppin, and Po-Sen Huang. 2021. Challenges in detoxifying language models. *arXiv preprint arXiv:2109.07445*.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

A Details of Experimental Setup

Parameter Setting and Training. Apart from the brief description in §5, we give more experimental details here. The DistilBERT we used for LEDGAR contains 6 transformer encoder layers each with 768 dimensions and 3072-dimensional feed-forward networks, resulting in 67M parameters. The transformer models used for IWSLT and PersonaChat are of the same size, i.e., containing 6 encoder and decoder layers each with 512 dimensions and 1024-dimensional feed-forward networks, with a total parameter amount of 91M. For the LSTM and BART-Base models we use in §6.4, the model sizes are 40M and 251M, respectively. The LSTM model contains 2 layers of encoder and decoder respectively, with 512 hidden size. The BART-Base model has 6 layers of 768-dimensional encoder and decoder, where we follow Lewis et al. (2020) to add new sets of encoder parameters before the pretrained BART encoder. This results in total 10 encoder layers (i.e., we add 4 layers).

We use one NVIDIA RTX 3090 GPU to train our model. When training the original model, the batch size is selected from $\{16, 32, 64\}$, and the final choices are 32 for LEDGAR and IWSLT, and 16 for PersonaChat, with an update frequency of 8. Learning rate is selected in $\{$1e-3, 5e-4, 2e-4, 5e-5, 2e-5$\}$, and we use 5e-5 for LEDGAR, 5e-4 for IWSLT, and 2e-4 for PersonaChat, respectively. Dropout strategy (Srivastava et al., 2014) with dropout rate selected in $\{0.1, 0.2, 0.3\}$ (the final choice is 0.1 for LEDGAR, and 0.3 for IWSLT and PersonaChat)

Removal	Source	Target	ORIGINAL	RETRAIN	KGA
become	Alle ihre Stimmen werden lauter und lauter, aber sie repräsentieren uns nicht.	Every one of them becomes a louder and louder voice, but they don't represent us.	All their voices become louder and louder, but they don't represent us.	All of their voices are getting louder and louder, but they don't represent us.	All their voices are louder and louder, but they don't represent us.
become	und diese Koordination riskiert, noch schwieriger zu werden mit der Einführung von Cyberwaffen.	And this coordination may become even trickier with the introduction of cyber weapons.	And this coordination may become even more difficult to become with the introduction of cyber weapons.	And this coordination risk to be even more difficult with the introduction of cyber weapons.	And that coordination, even harder to get into cyber weapons.
become	Anstelle des Treffens besserer Entscheidungen, werden wir von der Auswahl überwältigt manchmal macht sie uns sogar Angst.	Instead of making better choices, we become overwhelmed by choice, sometimes even afraid of it.	Instead of making better choices, we'll be overwhelmed by choice, sometimes even afraid.	Instead of meeting better decisions, we get overwhelmed by choice, sometimes it makes us fearful.	Instead of the meeting of better choices, we're even afraid of choice.
fresh	Wir reden hier über gute, frische Lebensmittel, die in unglaublichem Ausmaß verschwendet werden.	We're talking about good, fresh food that is being wasted on a colossal scale.	We're talking about good, fresh food that's being used in incredible scale.	We're talking about good, new food that's used in incredible scale.	We're talking about good foods that's going to be used in the incredible order of scale.
fresh	Wir brauchen einen neuen Standard für ordentliches frisches Essen für eure Kinder. Ja?	There needs to be a new standard of fresh, proper food for your children. Yeah?	We need a new standard for decent fresh food for your kids. Yes?	We need a new standard for proper new food for your kids. Right?	We need a new set of clean food for your kids. Yes?
fresh	Ich glaube dass hier zwei frische Ideen drin sind – zwei.	Well, I think there are two fresh things here – two fresh things.	I think there's two freshest ideas in here – two fresh water.	I think there are two new ideas in it – two.	I believe that there's two new ideas – two.
energy	Sie werden unübertroffene Vitalität und Energie gewinnen.	You'll have unsurpassed vitality and energy.	They're unsurprising vitality and energy.	They're won't win overblown vitality and power.	It's become overconducted vitality and power.
energy	Also habe ich gedacht, wie wir die Energiekrise in diesem Land bewältigen können?	And so I thought, how could we address the energy crisis in this country?	So I thought, how do we deal with the energy crisis in this country?	So I thought, how can we deal with the power crisis in this country?	So I thought, how do we deal with the crisis in this country?
energy	Energiepflanzen liefern ein halbes Watt pro Quadratmeter in europäischem Klima.	Energy crops deliver half a watt per square meter in European climates.	Energy crops deliver half a watt per square meter in European climates.	Power plants provide half watts per square meter in the European climate.	power plants deliver half a watt per square meter in European climates.

Table 6: Translation cases before and after unlearning from IWSLT dataset. After removing all training instances containing specific words (in red), the unlearned models tend to generate alternatives (in blue) with similar meanings to maintain consistency in terms of the whole sentence. For example, after removing all instances containing "energy" from original training set, the unlearned model generates "power" in corresponding positions.

and L_2 regularization with 0.0001 effect value are used to alleviate overfitting. During inference in generation tasks, the beam size is set to 5. All the above hyper-parameters are selected based on the performance of validation set.

Unlearning Setting. The removal numbers are set to 100 instances for LEDGAR and IWSLT, and 10 conversations (about 100 instances) for PersonaChat unless otherwise noted. We set the stopping hyper-parameter σ to 0.1.

B More Translation Unlearning Cases

Table 6 shows more cases when deleting all instances containing specific words, including "become" (verb), "fresh" (adjective), and "energy" (noun). We can find that unlearned models (i.e., RETRAIN and KGA) tend to generate alternatives with similar meanings regardless of the part of speech.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations

☑ A2. Did you discuss any potential risks of your work?
Ethics Statement

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract, Introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

5

☑ B1. Did you cite the creators of artifacts you used?
5

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
5

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
5

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
5

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
5

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
5

C ☑ Did you run computational experiments?

6

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
5, Appendix

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
6, Appendix

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.