5-2022

An empirical study of memorization in NLP

Xiaosen ZHENG
Singapore Management University, xszheng.2020@phdcs.smu.edu.sg

Jing JIANG
Singapore Management University, jingjiang@smu.edu.sg

Citation

ZHENG, Xiaosen and JIANG, Jing. An empirical study of memorization in NLP. (2022). *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, Dublin, Ireland, 2022 May 22 - 27*. 6265-6278.

Available at: https://ink.library.smu.edu.sg/sis_research/7705

An Empirical Study of Memorization in NLP

Xiaosen Zheng
Singapore Management University
xszheng.2020@phdcs.smu.edu.sg

Jing Jiang
Singapore Management University
jingjiang@smu.edu.sg

Abstract

A recent study by Feldman (2020) proposed a long-tail theory to explain the memorization behavior of deep learning models. However, memorization has not been empirically verified in the context of NLP, a gap addressed by this work. In this paper, we use three different NLP tasks to check if the long-tail theory holds. Our experiments demonstrate that top-ranked memorized training instances are likely atypical, and removing the top-memorized training instances leads to a more serious drop in test accuracy compared with removing training instances randomly. Furthermore, we develop an attribution method to better understand why a training instance is memorized. We empirically show that our memorization attribution method is faithful and share our interesting finding that the top-memorized parts of a training instance tend to be features negatively correlated with the class label.

1 Introduction

In recent years, there has been an increasing amount of interest in the machine learning community to understand the *memorization* behaviour of deep neural network models. Studies have shown that deep learning models often have sufficient capacities to "memorize" training examples (Zhang et al., 2017; Arpit et al., 2017). A number of recent studies tried to understand how memorization helps generalization (Chatterjee, 2018; Feldman, 2020; Montanari and Zhong, 2020; Khandelwal et al., 2020, 2021)

In NLP, memorization of training examples by deep learning models is also often observed (Li and Wisniewski, 2021; Lewis et al., 2021; Raunak et al., 2021), and existing studies usually see memorization as something that hinders generalization. For example, Elangovan et al. (2021) tried to measure the amount of "data leakage" in NLP datasets in order to assess a model's ability to memorize vs. its ability to generalize.

However, recently Feldman (2020) proposed a long-tail theory, which states that memorization is necessary for generalization if the data follows a long-tail distribution. This theory was later empirically validated by Feldman and Zhang (2020), but their validation was done in only the computer vision domain. It is therefore interesting and useful for us to study whether the long-tail theory also holds in NLP; such validation would help us better understand the utility of memorization in the context of NLP.

The long-tail theory states that if the training data form a long-tail distribution, where there are many small "sub-populations" that are atypical instances, and if these small sub-populations are also present in the test data, then memorizing these atypical instances helps the model generalize to the test data. In order to validate this long-tail theory in the context of NLP, we follow the experiments and analyses on image classification done by Feldman and Zhang (2020). Specifically, we aim to answer the following questions in this paper: (1) On a few typical NLP tasks, are the training instances memorized by deep learning models indeed *atypical* instances? (2) Does memorizing these training instances lead to *lower generalization error* on the test instances?

In addition, observing that it is not always straightforward to understand why a training instance is being memorized, we study the following novel research question: (3) Can we provide some explanation about why a training instances is memorized? To be more specific, can we attribute the memorization score of a training instance to its individual tokens such that we can quantify which tokens require the most memorization by the model?

To answer these research questions, we first adopt self-influence (Koh and Liang, 2017) as our memorization scoring function. Compared with the estimator proposed by Feldman and Zhang (2020), our self-influence function is also theoretically mo-

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 6265 - 6278
May 22-27, 2022 ©2022 Association for Computational Linguistics

tivated but has the advantage that it is easy for us to derive a memorization attribution method for the third research question above. We present the self-influence function in Section 2.1, and in Section 2.2, we present our novel memorization attribution method. We conduct experiments on three NLP tasks: sentiment classification, natural language inference (NLI) and text classification.

Our experiments and analyses demonstrate that *the training instances with the highest memorization scores tend to be atypical*, at least on sentiment classification and NLI. On all three tasks, we find that removing the top-memorized training instances results in significantly dropped test performance, and the drop is markedly higher compared with removing a random subset of training instances. We also evaluate our memorization attribution method and find that our method can indeed identify input tokens that require the most memorization. Finally, we apply our memorization attribution method to sentiment classification and to an image classification dataset, and we share the interesting finding that the highly-memorized input features tend to be those that are negatively correlated with the class labels. Our code and data are available at `https://github.com/xszheng2020/memorization`.

2 Our Approach

To validate the long-tail theory in the context NLP, let us first review the main claims of the theory. First, the long-tail theory hypothesizes that training instances with the same class label has a long-tail distribution, with instances at the tail end being those atypical instances that need to be memorized. To verify this assumption, we first identify those training instances that are memorized by a trained deep learning model and then check if they are indeed atypical. Specifically, we follow Feldman and Zhang (2020) and adopt "self-influence" to measure memorization, but we use the influence function proposed by Koh and Liang (2017) to define self-influence. Second, the long tail theory states that memorization of the atypical training instances leads to lower generalization error, because the atypical training instances belong to subpopulations that also have presence in the test data. To verify this statement, we check whether removing the memorized training instances would lead to more significant performance drop on the test data than removing a random sample of training instances.

It is worth noting that the approach outlined above follows the experiments conducted by Feldman and Zhang (2020) to validate the long tail theory on image classification.

Furthermore, we want to pinpoint which parts of a memorized instance are most critical for memorization. In other words, since each training instance is assigned a memorization score, can we attribute the memorization score to different parts of the input of this instance? This presumably can help us better understand which parts of the input need to be memorized the most. We follow the idea from Integrated Gradients (IG) (Sundararajan et al., 2017) and derive a formula to compute memorization attribution.

2.1 Memorization: Self-Influence

The high level idea of Feldman (2020) to define memorization is that memorization measures how the prediction on a training instance $z = (x, y)$ (where x is the observation and y is the label) changes when z is removed from the training data. This notion is closely related to the influence function defined by Koh and Liang (2017), which measures how much the loss at a test point z_{test} is influenced by a slight upweighting of a training instance z in the training loss function. While influence function is generally used to measure the influence of a training instance on a *test* instance, if we use it to measure the influence of a training instance on *itself*, i.e., to measure "self-influence," then this self-influence corresponds to the general notion of memorization defined by Feldman (2020).

Adopting the influence function defined by Koh and Liang (2017), we define the memorization score for a training instance z as follows:

$$\mathcal{M}_{\text{remove}}(z) \overset{\text{def}}{=} - \frac{dP(y|x;\hat{\theta}_{\epsilon,-z})}{d\epsilon}\bigg|_{\epsilon=0}, \quad (1)$$

where $\hat{\theta}_{\epsilon,-z}$ represents the parameters of the model trained with the instance z down-weighted by ϵ, $P(y|x;\theta)$ is the conditional probability using θ. Thus $\mathcal{M}_{\text{remove}}(z)$ is the amount of change of $P(y|x;\theta)$ when the instance z is down-weighted by a small amount ϵ.

After several steps of derivation (details to be given in Appendix A), the computation of Eqn 1 follows the following formula:

$$\mathcal{M}_{\text{remove}}(z) = -\nabla_{\theta} P(y|x;\hat{\theta})^{\top} H_{\hat{\theta}}^{-1} \nabla_{\theta} L(z, \hat{\theta}), \quad (2)$$

where $\hat{\theta}$ is the parameters of the model trained with all instances, L is the loss function (cross entropy in our implementation) and $H_{\hat{\theta}} = \frac{1}{n}\sum_{i=1}^{n} \nabla_{\theta}^{2} L(z_i, \hat{\theta})$, where $(z_1, z_2, \dots, z_n)$ are the training instances.

2.2 Memorization Attribution

In order to better understand why an instance is memorized, we propose a fine-grained notion of memorization at "feature" level instead of instance level, i.e., to attribute the memorization score of an instance to its individual features. Our proposed memorization attribution method is general and can be applied to any input representation. For NLP tasks, this means we attribute the memorization score defined above to each token of the input sequence. For images, this would be to attribute the memorization scores to pixels.

For this memorization attribution, we borrow the idea from Integrated Gradients (IG) (Sundararajan et al., 2017), which is a gradient-based attribution method for understanding which parts of a test instance are more responsible for its prediction. In particular, the IG method requires an *uninformative* baseline input x' as a reference point. Similarly, here we also assume a baseline x'. This baseline is supposedly an instance that does not have any influence on any test instance, and in our implementation, we use an sequence of the same length as x but consisting of only the `[MASK]` tokens.

We first consider the influence of replacing $z = (x, y)$ with the baseline $z' = (x', y)$ (which is similar to perturbation-based influence from (Koh and Liang, 2017)):

$$\mathcal{M}_{\text{replace}}(z) \overset{\text{def}}{=} -\left.\frac{dP(y|x;\hat{\theta}_{\epsilon,z',-z})}{d\epsilon}\right|_{\epsilon=0}, \quad (3)$$

where $\hat{\theta}_{\epsilon,z',-z}$ represents the parameters resulting from moving ϵ mass from z to z', i.e., adding z' to the training data and giving it a weight of ϵ in the loss function while reducing the weight of the original z by ϵ. Thus $\mathcal{M}_{\text{replace}}(z)$ is the amount of change of $P(y|x;\theta)$ when a small amount ϵ of z is replaced by the uninformative z'.

It is worth pointing out that we can regard $\mathcal{M}_{\text{replace}}(z)$ as an alternative way of measuring the amount of memorization of z, similar to how perturbation-based influence is an alternative way of measuring influence in (Koh and Liang, 2017).

With similar derivation steps, the computation of Eqn 3 is as follows:

$$\mathcal{M}_{\text{replace}}(z) = -s^{\top}\left(\nabla_{\theta} L(z, \hat{\theta}) - \nabla_{\theta} L(z', \hat{\theta})\right), \quad (4)$$

where $s = H_{\hat{\theta}}^{-1} \nabla_{\theta} P(y|x;\hat{\theta})$. (For more details, please refer to Appendix B.)

The advantage of using this alternative measure of memorization is that $\mathcal{M}_{\text{replace}}(z)$ can be decomposed into a linear combination of scores, each corresponding to a single token in the input sequence. For NLP applications, the input x usually corresponds to an embedding matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$ (where N is the number of tokens and d is the embedding dimensions). We can show that

$$\mathcal{M}_{\text{replace}}(z) = -\sum_{t=1}^{N}\sum_{l=1}^{d} r_{t,l}(\mathbf{X}_{t,l} - \mathbf{X}'_{t,l}), \quad (5)$$

where $r = \left[\int_{\alpha=0}^{1} \frac{dg(\mathbf{X}'+\alpha(\mathbf{X}-\mathbf{X}'))}{dx} d\alpha\right] s$ and $g(\mathbf{X}) = \nabla_{\theta} L((\mathbf{X}, y), \hat{\theta})$, which can be efficiently computed by the hessian-vector product (Pearlmutter, 1994). For more details, please refer to Appendix B.

The memorization attribution of the t-th token is thus given by $-\sum_{l=1}^{d} r_{t,l} \times (\mathbf{X}_{t,l} - \mathbf{X}'_{t,l})$.

3 Experiments

With the memorization score defined in Eqn 2 and the memorization attribution score defined in Eqn 5, we now conduct experiments to answer the three research questions raised in Section 1.

3.1 Experiment Settings

We conduct our experiments on the following three datasets:

SST-2 (Socher et al., 2013): This is a dataset for sentence-level binary (positive vs. negative) sentiment classification. It consists of 6,920 training instances, 872 development instances and 1,821 test instances.

SNLI (MacCartney and Manning, 2008): This is a dataset for natural language inference, which aims to predict the entailment relation (contradiction, neutral or entailment) between a premise and a hypothesis. We combine the *contradiction* and *neutral* classes into a single *non-entailment* class, and randomly sample 10k training instances, 6,658 development instances and 6,736 test instances.

Yahoo! Answers (Zhang et al., 2015): This is a collection of question-answer pairs categorized into 10 topic-based classes. We randomly sample 10k training instances, 10k development instances and 10k test examples.

In addition, we also use **CIFAR-10** (Krizhevsky et al., 2009), which is a dataset for 10-class image classification. We randomly sample 10k training instances, 5k development instances and 10k test instances. For some tasks, we down-sample the training set because influence function is known to be expensive to compute.

For all NLP tasks, we adopt the pre-trained Distill-BERT model (Sanh et al., 2019) that consists of 6 transformer layers, where each layer consists of 12 attention heads. We use the final hidden state of the `[CLS]` token for classification.[1] For CIFAR-10, we extract visual grid features using a pre-trained ResNet50 (He et al., 2016) first and then train a MLP classifier on top of that.

We use the SGD optimizer, setting the learning rate, momentum and batch size to 0.01, 0.9 and 32, respectively. We tune other hyper-parameters on the development set manually.

Although influence function is model-dependent and therefore models trained with different random seeds may produce different memorization scores for the same training instance, we found that in practice, ranking training instances based on memorization scores obtained from models trained by different random seeds produces similar rankings across different models. Thus, we only consider a single model checkpoint for computing our self-influence based memorization scores in the following experiments. (See Appendix C for the exact description.) For memorization attribution, the number of Riemann Sum steps is set to be 50.

3.2 Checking Memorized Instances

Group	Negative	Positive
Top-10%	35.80	74.00
All	23.24	86.39
Bottom-10%	14.92	94.52

Table 1: The average percentage of *positive phrases* over (1) the top-10% memorized positive/negative instances, (2) all positive/negative instances, and (3) the bottom-10% memorized positive/negative instances.

In the first set of experiments, we use our self-influence-based memorization scoring function as defined in Eqn. 1 to rank the training instances.

Our goal is to check if the top-memorized instances are indeed atypical instances. However, it is difficult to measure the typicality of instances. We note that in the prior work (Feldman and Zhang, 2020) where the authors tried to validate the long-tail theory on computer vision datasets, there was not any quantitative experiment, and the authors relied only on qualitative analysis (i.e., manual inspection of the top-ranked instances) to show that memorized instances tend to be atypical. In our experiments, we perform two kinds of checking: (1) First, we adopt qualitative evaluation as Feldman and Zhang (2020) did on both SST-2 and SNLI. For Yahoo! Answers, however, because each instance contains a long document, it is not easy for humans to judge whether or not an instance is atypical. (2) Second, we define quantitative measures of typicality on sentiment analysis because annotations are available on this dataset and these annotations allow us to define some form of typicality.

SST-2

For SST-2, we judge whether or not the top-ranked memorized instances are atypical in two ways: (1) The first is based on a heuristic metric. We check the percentage of positive phrases in an instance, where phrase-level sentiment polarity labels are from the annotations provided by SST-2. Intuitively, a typical positive sentence should have a relatively high percentage of positive phrases and a typical negative sentence should have a relatively low percentage of positive phrases. We collect such statistics from SST-2 based on the phrase-level annotations and found that this is to a large extent true. For example, more than 75% of positive sentences have at least 78.31% of positive phrases and more than 75% of negative sentences have at most 35.73% of positive phrases. (See Appendix D for details.) Therefore, by checking the percentage of positive phrases inside a positive or negative instance, we can in a way judge whether that instance is typical or atypical. When calculating the percentage of positive phrases inside a sentence, we apply Laplace smoothing. (2) We also manually inspect the top-ranked and bottom-ranked training instances based on the memorization scores and use our human knowledge to judge whether the top-ranked ones are atypical while the bottom-ranked ones are typical.

[1] Following Han et al. (2020); Guo et al. (2021), we "freeze" the word embedding layer and the first 4 transformer layers, only fine-tuning the last 2 transformer layers and the final projection layer because of the computation limits.

Negative		Positive	
Content	**Mem**	**Content**	**Mem**
Starts out with tremendous promise, introducing an intriguing and alluring premise, only to fall prey to a boatload of screenwriting cliches that sink it faster than a leaky freighter	14.83	The director, Mark Pellington, does a terrific job conjuring up a sinister, menacing atmosphere though unfortunately all the story gives us is flashing red lights, a rattling noise, and a bump on the head	14.28
Mr. Wollter and Ms. Seldhal give strong and convincing performances, but neither reaches into the deepest recesses of the character to unearth the quaking essence of passion, grief and fear	13.65	This is a fascinating film because there is no clear-cut hero and no all-out villain	14.18
This is a monumental achievement in practically every facet of inept filmmaking: joyless, idiotic, annoying, heavy-handed visually atrocious, and often downright creepy	11.01	The film is reasonably entertaining, though it begins to drag two-thirds through, when the melodramatic aspects start to overtake the comedy	11.04
Sadly, Full Frontal plays like the work of a dilettante	0.00	The large-format film is well suited to capture these musicians in full regalia and the incredible IMAX sound system lets you feel the beat down to your toes	0.00
A mess	0.00	P.T. Anderson understands the grandness of romance and how love is the great equalizer that can calm us of our daily ills and bring out joys in our lives that we never knew were possible	0.00
The images lack contrast, are murky and are frequently too dark to be decipherable	0.00	together writer-director Danny Verete's three tales comprise a powerful and reasonably fulfilling gestalt	0.00

Table 2: Top-3 and Bottom-3 memorized training examples from the SST-2 task. Note that there are many examples having zero memorization score, we randomly sample 3 out of them.

Non-Entail		Entail	
Content	**Mem**	**Content**	**Mem**
P: A man in a bright pastel blue overcoat plays a unique instrument by the corner of a building with a sign propped against a bag in front of him **H:** A man plays a guitar outside	18.85	**P:** An older man in a white shirt is playing a keyboard **H:** A man is playing the piano	23.24
P: A young boy in a yellow rash guard is walking on the shore carrying a surfboard **H:** A boy is walking on the boardwalk	17.51	**P:** A woman in a white and light green jacket and another woman in a purple shirt , both wearing hats , sit at a table watching a cooking fire **H:** A woman in a white and light green jacket	18.94
P: Someone wearing a blue shirt is riding a bike with a child ' s seat on the front of it **H:** A person is riding a bike on the street	15.52	**P:** A man sits on a folding chair outside while listening to music on his iPod **H:** There is a man on a chair listening to music on an mp3 player	18.89
P: A brunette woman does a wheelie on a white bicycle with purple tires **H:** A woman rides her motorcycle to town	0.00	**P:** A married man is taking pictures while standing in a crowd of people **H:** There are people in a crowd	0.00
P: A baseball player hitting a home run **H:** The cat eats sheep	0.00	**P:** A man recreates a joust from mid - evil times **H:** A person created something	0.00
P: A child in a vest and hat is posing for a picture **H:** A child is eating his lunch	0.00	**P:** A boy is wearing a red towel standing on the beach **H:** A person is at the beach	0.00

Table 3: Top-3 and Bottom-3 memorized training examples from the SNLI task. Note that there are many examples having zero memorization score, we randomly sample 3 out of them.

Table 1 shows the average percentage of positive phrases in the top-10% of the memorized positive (or negative) training instances and the bottom-10% of the memorized positive (or negative) training instances. As a reference point, we also show the average percentage over all positive (or negative) training instances. We can see that the top-10% memorized instances indeed are atypical. Specifically, those negative sentences with high memorization scores have a high percentage of positive phrases on average (35.80%), clearly higher than the average percentage of positive phrases of all negative instances (23.24%). This makes the top-memorized negative instances very different from typical negative instances. On the other hand, the bottom-10% negative instances (i.e., those instances that are not memorized) have clearly much lower percentage of positive phrases (14.92%), which is what we expect for typical negative instances. Similar observations can be made with the positive training instances. Overall, the results in Table 1 suggest that indeed the top-memorized training instances in SST-2 are atypical.

Next, we manually inspect the top-ranked and bottom-ranked training instances of SST-2 in Table 2. We can see that the top-ranked memorized instances tend to express their overall opinions in an indirect way. These sentences often contain a contrast between positive and negative opinions. We therefore believe that they are atypical for sentiment classification. On the other hand, the bottom-ranked instances, i.e., those with 0 memorization scores, tend to directly expression their opinions with strong opinion phrases, and we believe these represent common instances.

SNLI

For the task of natural language inference, it is hard to come up with a heuristic metric like the one used for sentiment classification. We therefore focus on manual inspection of the top-ranked and bottom-ranked training instances. In Table 3 we show the top-3 and bottom-3 memorized training instances from SNLI. We can see from the table that in the top-ranked memorized non-entailment instances, the hypothesis tends to be much shorter than the premise and there tends to be no obvious contradiction. In contrast, the bottom-ranked non-entailment instances tend to be contradictions where there are obvious contradictory words/phrases in the premise and the hypothesis, such as “bicycle” vs. “motorcycle,” “player” vs. “cat” and “posing for a picture” vs. “eating his lunch.” We hypothesize that the top-ranked non-entailment instances are atypical because they do not have obvious signals of non-entailment such as the contradictory word pairs we see in the bottom-ranked non-entailment instances. For entailment cases, we find that the top-ranked instances often contain word pairs that are synonyms but are rare in the training data. For example, we find that the word pair “keyboard” and “piano” appears only two times in the training data, which implies that this instance is an atypical example. Similarly, we find that the word/phrase pair “iPod” and “mp3 player” appear only once in the training data. On the other hand, the bottom-ranked entailment instances tend to be those where the hypothesis contains less information than the premise, which may be a common type of entailment instances.

3.3 Marginal Utility of Memorized Instances

In the second set of experiments, we check whether memorizing those training instances with the highest memorization scores leads to better performance on the unseen test data. To do so, we compare the performance of the model on test data when top-ranked memorized training instances are removed during training versus the performance when the same number of *randomly* selected training instances are removed. If memorization is beneficial for the test data, then we would expect to see larger performance drop when top-ranked memorized training instances are removed than when random training instances are removed. Therefore, the amount of performance drop represents the marginal effect of the memorized instances on the test accuracy. We show the test accuracy in Figure 1 when $X\%$ of the training instances are removed, where we set X to a few different values. We re-train the model 5 times and show the average test accuracy as well as the standard deviation. We also show the lowest absolute memorization score of the top-$X\%$ of training instances in Figure 1. For reference, here we also use CIFAR-10 to verify that our self-influence estimation using the influence function works similarly to the influence estimator used by Feldman and Zhang (2020).

We can observe the following from Figure 1: (1) On CIFAR-10 (Figure 1(d)), we see that clearly the test accuracy drops more significantly when top-ranked memorized training instances instead of random training instances are removed. Because Feldman and Zhang (2020) reported the same observation, this suggests that our memorization score based on the influence function proposed by Koh and Liang (2017) works similarly to the memorization estimator used by Feldman and Zhang (2020). This verifies the reliability of our memorization scoring function. (2) On SST-2, Yahoo! Answers and SNLI, we can see that consistently when the same percentage of training instances are removed, removing top-ranked memorized instances has a clearly bigger impact on the test accuracy compared with removing random instances. For example, on SST-2, the marginal utility of the top-30% memorized training example is about 1.44 percentage points (vs. 0.70 percentage points for random subset of 30% of training examples).

This verifies that on SST-2, Yahoo! Answers and SNLI, memorizing those training instances could help improve the performance on the test data.

3.4 Evaluating Memorization Attribution

In this section, we evaluate whether our memorization attribution method is faithful, i.e., whether it indeed picks up tokens that have higher self-influence.

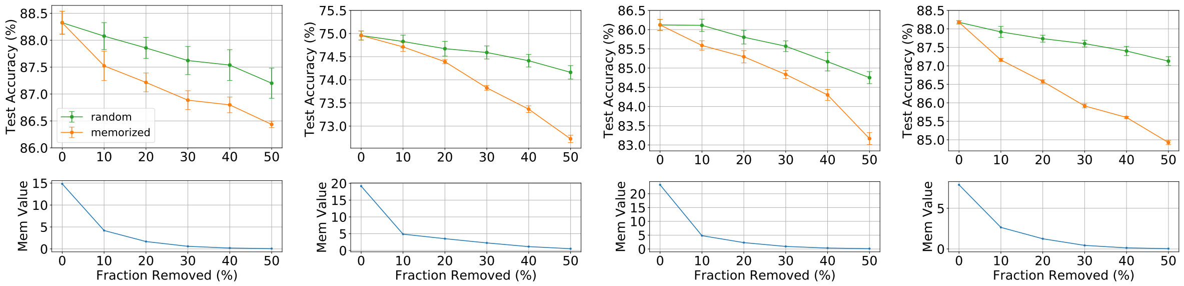
Intuitively, if the memorization attribution method detects those memorized tokens in a training instance faithfully, then removing these tokens in that instance should result in a lower influence $\mathcal{I}$ of the perturbed instance on its original form (details to be given in Appendix A). We therefore define a metric called Reduction Rate as follows:

$$\frac{1}{|\mathcal{Z}|}\sum_{z\in\mathcal{Z}}\frac{\mathcal{I}(z,z)-\mathcal{I}(z^{\setminus\text{attr}},z)}{\mathcal{I}(z,z)}, \tag{6}$$

where $\mathcal{Z}$ is the set of top memorized training instances and $z^{\setminus\text{attr}}$ is the perturbed input where the top-$k\%$ memorized tokens are replace by the baseline token `[MASK]`. We can see that this Reduction Rate measures how much self-influence has been reduced after the top-memorized tokens are replaced with `[MASK]`.[2]

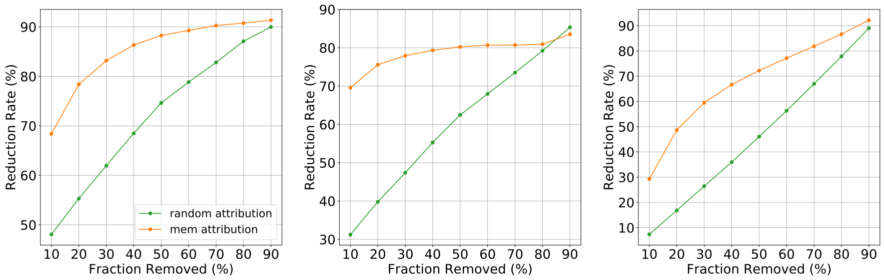
Figure 2 demonstrates the significant effect of the removal of the top-memorized tokens from the top-memorized training instances. One could ask whether this effect is solely due to the input perturbation. To answer this question we include in the

[2] We consider only top-10% memorized instances due to computation constraints.



(a) SST-2 (b) Yahoo! Answers (c) SNLI (d) CIFAR-10

Figure 1: For each dataset, the top figure shows the test accuracy after we remove the top-$X\%$ memorized training instances or the same number of randomly selected training instances. The test accuracy is averaged over 5 runs of retraining with different random seeds, and standard deviation is shown with the bars. The bottom figure shows the lowest memorization score of the top-$X\%$ of the memorized training instances.



(a) SST-2 (b) Yahoo! Answers (c) CIFAR-10

Figure 2: For each dataset, the top figure shows the reduction rate of removing the top-$k\%$ memorized tokens and of removing the same number of randomly selected training tokens.

comparison the reduction rate of random attribution, i.e., we randomly remove some tokens from the training instances. We can see that removing tokens picked up by our memorization attribution method results in a much larger Reduction Rate until almost 90% of the tokens are removed. This result suggests that our memorization attribution method can indeed identify those tokens in a training instance that have high self-influence on that instance.

3.5 Examples of Memorization Attribution

To better understand why certain training instances are memorized, we apply our memorization attribution method to SST-2, Yahoo! Answers and CIFAR-10. We do not discuss our memorization attribution method applied to the NLI task because we find that it is not easy to interpret the results. In some other studies (e.g., Han et al. (2020)), people have also reported different behaviours of NLI from tasks relying on shallow features such as sentiment classification and topic-based text classification.

We find that on SST-2, Yahoo! Answers and CIFAR-10, in most cases our memorization attributions are easy to be interpreted by humans. In particular, without any cherry-picking, we select those instances with the highest memorization scores to present. We find that interestingly, for both SST-2 and CIFAR-10, the trained deep learning model tends to memorize those parts of an instance that are negatively correlated with the class label of that instance, as shown in Table 4 and Figure 3.[3] On SST-2, for example, the model needs to memorize positive phrases such as "tremendous promise" and "intriguing and alluring" that show up in an overall negative instance. On CIFAR-10, we observe that for images that are easily mis-classified, the model memorizes those pixels that are associated with the wrong class label, or in other words, pixels that are negatively correlated with the correct class label. For example, the "cat" image shown in Figure 3 looks like a frog. The model memorizes those pixels (shown in red) around the tummy of the cat

[3] For Yahoo! Answers, because each instance is long, due to the space limit, we show the memorization attributions in the Appendix E.

Content	Label
starts out with tremendous promise introducing an intriguing and alluring premise only to fall prey to a boatload of screenwriting cliches that sink it faster than a leaky freighter	Neg
mr wollter and ms seldhal give strong and convincing performances but neither reaches into the deepest recesses of the character to unearth the quaking essence of passion grief and fear	Neg
this is a monumental achievement in practically every facet of inept filmmaking joyless idiotic annoying heavy handed visually atrocious and often downright creepy	Neg
the director mark pellington does a terrific job conjuring up a sinister menacing atmosphere though unfortunately all the story gives us is flashing red lights a rattling noise and a bump on the head	Pos
this is a fascinating film because there is no clear cut hero and no all out villain	Pos
the film is reasonably entertaining though it begins to drag two thirds through when the melodramatic aspects start to overtake the comedy	Pos

Table 4: The top-3 memorized training instances for each class from SST-2. Highlighted words are those with high attribution values (red for positive memorization attribution and blue for negative memorization attribution) as computed by our memorization attribution method.

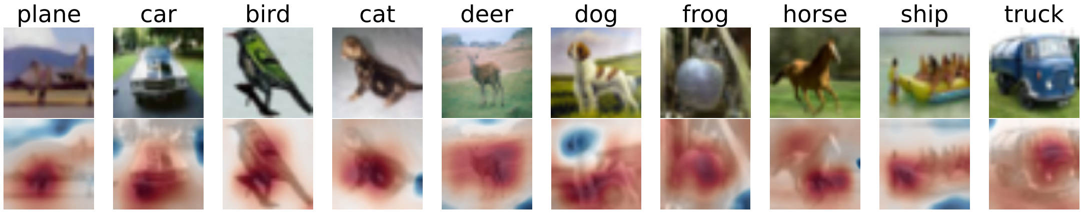


Figure 3: The top-1 memorized training instance for each class from CIFAR-10. Highlighted patches are those having high attribution values (red for positive memorization attribution and blue for negative memorization attribution) as computed by our memorization attribution method.

because those pixels make the image look like a frog image. Similarly, in the “dog” image, which looks like a horse, the memorized pixels (shown in red) are around the body of the dog, and these pixels make the image look like a horse image. On the other hand, the dog’s head in this image, which is a typical dog’s head, has negative memorization attribution scores, which means it does not need to be memorized.

Given the interesting results above, we believe that model developers can gain insights about what a model finds hard to learn from other training instances (and thus has to memorize), and model developers can subsequently take actions like upweighting memorized instances or collecting similar data to improve the performance on certain subpopulations if desired.

4 Related Work

The long-tail theory: The long-tail theory proposed by Feldman (2020) is relatively new and has not been systematically validated in NLP. Our work is the first to empirically check the validity of this theory on NLP tasks. Raunak et al. (2021) used the long-tail theory to explain hallucinations under source perturbations in Neural Machine Translation. They assume the theory holds in NMT rather than validating the theory itself as we do. Kong and Chaudhuri (2021) investigated the memorization phenomenon for Variational Auto-Encoder also via self-influence.

Memorization vs. generalization: It is well-known that deep learning models possess strong capabilities to memorize training instances (Zhang et al., 2017; Arpit et al., 2017). In the context of NLP, Li and Wisniewski (2021) showed that BERT is more likely to memorize shallow patterns from the training data rather than uncover abstract properties. Some recent work has tried to combine interpolation methods with deep learning methods to generalize via memorization (Khandelwal et al., 2020, 2021). However, previous work using interpolation methods did not explain why memorization is necessary in the first place. Our work follows the long-tail theory that views memorization as beneficial to generalization when the data follows a certain type of long-tail distribution. There has also been some work studying “forgetting,” which is related to memorization (Toneva et al., 2018; Yaghoobzadeh et al., 2021). However, in this paper we do not study this “forgetting” phenomenon.

Influence functions: Influence functions have been studied for large-scale deep neural networks by Koh and Liang (2017) and gained much attention in recent years. In the context of NLP, Han et al. (2020) explored the usage of influence functions to explain model predictions and unveil data artifacts. Meng et al. (2020) proposed a combination of gradient-based methods and influence functions to examine training history and test stimuli simultaneously. Our work, however, adopts influence function as a tool to measure memorization.

5 Conclusions

In this paper, we empirically examine a recently proposed long-tail theory in the context of NLP. We use sentiment classification, natural language inference and text classification to check the validity of the long-tail theory in NLP. We also propose a memorization attribution method to reveal which parts of an instance are being memorized. There are two major takeaway messages: (1) Our experiments empirically validated the long-tail theory on the three NLP datasets, showing that memorization is important for generalization, offers an alternative view and helps NLP researchers to see the value of memorization. (2) Our attribution method can be a tool to help model developers better understand the memorization behaviours of a model and possibly further improve the model.

6 Ethical Considerations

Our work empirically validated the long-tail theory in the context of NLP, offering an alternative view to the relationship between memorization and generalization. This will help NLP researchers see the value of memorization. However, previous work showed that neural networks can be vulnerable to privacy attacks such as membership inference attacks because these models are able to memorize training instances, and sometimes sensitive private information may be contained in the training instances (Shokri et al., 2017; Zhang et al., 2017; Feldman and Zhang, 2020). Thus, there is a tradeoff between the accuracy of a model and the privacy of the data. In other words, although memorization can help reduce generalization error (as we showed in this paper), it also increases the vulnerability of the system to privacy attacks, which raises ethical concerns.

The computation of influence functions used in our work is massive because of the computation of inverting the hessian matrices. To reduce the computation costs, i.e., power consumption, we may adopt other influence estimators like TracIn (Pruthi et al., 2020), which is hessian-free and thus faster.

Acknowledgment

This research is supported by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 grant.

References

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. 2017. A closer look at memorization in deep networks. In *International Conference on Machine Learning*.

Satrajit Chatterjee. 2018. Learning and memorization. In *International Conference on Machine Learning*.

Aparna Elangovan, Jiayuan He, and Karin Verspoor. 2021. Memorization vs. generalization : Quantifying data leakage in NLP performance evaluation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1325–1335, Online. Association for Computational Linguistics.

Vitaly Feldman. 2020. Does learning require memorization? a short tale about a long tail. In *Annual ACM SIGACT Symposium on Theory of Computing*.

Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Advances in Neural Information Processing Systems*.

Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. 2021. FastIF: Scalable influence functions for efficient model interpretation and debugging. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10333–10350, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563, Online. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*.

Harold V Henderson and Shayle R Searle. 1981. On deriving the inverse of a sum of matrices. *Siam Review*.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. *International Conference on Learning Representations*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. *International Conference on Learning Representations*.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*.

Zhifeng Kong and Kamalika Chaudhuri. 2021. Understanding instance-based interpretability of variational auto-encoders. In *Advances in Neural Information Processing Systems*.

Alex Krizhevsky, Geoffrey Hinton, et al. 2009. *Learning multiple layers of features from tiny images*. Citeseer.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1000–1008, Online. Association for Computational Linguistics.

Bingzhi Li and Guillaume Wisniewski. 2021. Are neural networks extracting linguistic properties or memorizing training data? an observation with a multilingual probe for predicting tense. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3080–3089, Online. Association for Computational Linguistics.

Bill MacCartney and Christopher D. Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 521–528, Manchester, UK. Coling 2008 Organizing Committee.

Yuxian Meng, Chun Fan, Zijun Sun, Eduard Hovy, Fei Wu, and Jiwei Li. 2020. Pair the dots: Jointly examining training history and test stimuli for model interpretability. *arXiv preprint arXiv:2010.06943*.

Andrea Montanari and Yiqiao Zhong. 2020. The interpolation phase transition in neural networks: Memorization and generalization under lazy training. *arXiv preprint arXiv:2007.12826*.

Barak A Pearlmutter. 1994. Fast exact multiplication by the hessian. *Neural computation*.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*.

Vikas Raunak, Arul Menezes, and Marcin Junczys-Dowmunt. 2021. The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1172–1183, Online. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Workshop on Energy Efficient Machine Learning and Cognitive Computing, Advances in Neural Information Processing Systems*.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. 2018. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*.

Yadollah Yaghoobzadeh, Soroush Mehri, Remi Tachet des Combes, T. J. Hazen, and Alessandro Sordoni. 2021. Increasing robustness to spurious correlations using forgettable examples. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3319–3332, Online. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*.

A Derivation of the Memorization Scores

For clarity, here we repeat the derivation of Influence Functions by Koh and Liang (2017) and provide self-influence functions as its special case. Note that self-influence functions are used as our memorization scores.

Given training points $z_1, ..., z_n$, where $z_i = (x_i, y_i)$, x_i is the observation and y_i is the label, we train a predictor via minimizing the empirical risk $R(\theta) \overset{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$ to pick parameters $\theta \in \Theta$. I.e., the optimal parameters are obtained by $\hat{\theta} = \arg\min_{\theta \in \Theta} R(\theta)$. We assume that R is twice-differentiable and strongly convex.

i.e.,

$$H_{\hat{\theta}} \overset{\text{def}}{=} \nabla^2 R(\hat{\theta}) = \frac{1}{n}\sum_{i=1}^{n} \nabla_{\theta}^2 L(z_i, \hat{\theta}) \tag{7}$$

exists and is positive definite. This guarantees the existence of $H_{\hat{\theta}}^{-1}$, which we will use in the following derivation.

The high-level idea of Influence Functions is to approximate leave-one-out retraining, which corresponds to a removing operation, via computing the parameter change if z were up-weighted or down-weighted by some small amount ϵ.

If we up-weight the training point z, the perturbed parameters $\hat{\theta}_{\epsilon,z}$ can be written as

$$\hat{\theta}_{\epsilon,z} = \arg\min_{\theta \in \Theta} \left(R(\theta) + \epsilon L(z, \theta)\right). \tag{8}$$

Consider the parameter change $\Delta_\epsilon = \hat{\theta}_{\epsilon,z} - \hat{\theta}$, and note that, as $\hat{\theta}$ does not depend on ϵ, the quantity we want to compute can be written in terms of it:

$$\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon} = \frac{d\Delta_\epsilon}{d\epsilon}. \tag{9}$$

Since $\hat{\theta}_{\epsilon,z}$ is a minimizer of Eqn 8, let us examine its first-order optimality condition:

$$0 = \nabla R(\hat{\theta}_{\epsilon,z}) + \epsilon \nabla L(z, \hat{\theta}_{\epsilon,z}). \tag{10}$$

Let us define $f(\theta)$ to be $\nabla R(\theta) + \epsilon \nabla L(z, \theta)$.

Next, since $\hat{\theta}_{\epsilon,z} \to \hat{\theta}$ as $\epsilon \to 0$, we perform a Taylor expansion on $f(\hat{\theta}_{\epsilon,z})$. Given Taylor's Formula $f(\theta + \Delta\theta) = f(\theta) + f'(\theta)\Delta\theta + o(\Delta\theta)$, we have:

$$\begin{aligned} 0 &= f(\hat{\theta}_{\epsilon,z}) \\ &= f(\hat{\theta} + \Delta_\epsilon) \\ &\approx f(\hat{\theta}) + f'(\hat{\theta})\Delta_\epsilon \\ &= [\nabla R(\hat{\theta}) + \epsilon \nabla L(z, \hat{\theta})] \\ &\quad + [\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})]\Delta_\epsilon, \end{aligned} \tag{11}$$

where we have dropped the term $o(\|\Delta_\epsilon\|)$.

Solving for Δ_ϵ, we get $\Delta_\epsilon \approx -[\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})]^{-1}[\nabla R(\hat{\theta}) + \epsilon \nabla L(z, \hat{\theta})]$.

Since $\hat{\theta}$ minimizes R, we have $\nabla R(\hat{\theta}) = 0$. Then we have:

$$\Delta_\epsilon \approx -[\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})]^{-1} \epsilon \nabla L(z, \hat{\theta}). \tag{12}$$

Referring to (Henderson and Searle, 1981), we have:

$$\begin{aligned} (A + B)^{-1} &= (I + A^{-1}B)^{-1}A^{-1} \\ &= A^{-1} - A^{-1}B(I + A^{-1}B)^{-1}A^{-1} \\ &= A^{-1} - A^{-1}B(A + B)^{-1}, \end{aligned} \tag{13}$$

which only requires A and $A + B$ to be nonsingular matrix. As mentioned above, the matrices that we are considering are positive definite. The determinant of a positive definite matrix is always positive, so a positive definite matrix is always nonsingular.

Substituting $A = \nabla^2 R(\hat{\theta})$ and $B = \epsilon \nabla^2 L(z, \hat{\theta})$ and dropping $o(\epsilon)$ terms, we have

$$\Delta_\epsilon \approx -\nabla^2 R(\hat{\theta})^{-1} \nabla L(z, \hat{\theta})\epsilon. \tag{14}$$

Combining with Eqn 7 and Eqn 9, we conclude that:

$$\left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1} \nabla L(z, \hat{\theta}). \tag{15}$$

We instead down-weight the training point z to keep consistency with our memorization attribution method introduced later, the perturbed parameters $\hat{\theta}_{\epsilon,-z}$ can be written as

$$\hat{\theta}_{\epsilon,-z} = \arg\min_{\theta \in \Theta} \left(R(\theta) - \epsilon L(z, \theta)\right). \tag{16}$$

It is easy to see that

$$\left.\frac{d\hat{\theta}_{\epsilon,-z}}{d\epsilon}\right|_{\epsilon=0} = H_{\hat{\theta}}^{-1} \nabla L(z, \hat{\theta}). \tag{17}$$

Next, we apply the chain rule to measure how down-weighting z changes functions of $\hat{\theta}$.

$$\begin{aligned} \mathcal{I}(z, z_{\text{test}}) &\overset{\text{def}}{=} \left.\frac{dF(y_{\text{test}}, x_{\text{test}}; \hat{\theta}_{\epsilon,-z})}{d\epsilon}\right|_{\epsilon=0} \\ &= \nabla_\theta F(y_{\text{test}}, x_{\text{test}}; \hat{\theta})^\top \left.\frac{d\hat{\theta}_{\epsilon,-z}}{d\epsilon}\right|_{\epsilon=0} \\ &= \nabla_\theta F(y_{\text{test}}, x_{\text{test}}; \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta}), \end{aligned} \tag{18}$$

where F is usually the loss function.

While influence function is generally used to measure the influence of a training instance on a test instance, if we use it to measure the influence of a training instance on itself, i.e., to measure self-influence, then this self-influence corresponds to the general notion of memorization defined by Feldman (2020); Feldman and Zhang (2020). Based on this notion, we set F as the negative estimated conditional probability $-P(y|x;\theta)$ and define the memorization score for a training instance z as follows:

$$\begin{aligned}\mathcal{M}_{\text{remove}}(z) &\overset{\text{def}}{=} -\frac{dP(y|x;\hat{\theta}_{\epsilon,-z})}{d\epsilon}\bigg|_{\epsilon=0} \\ &= -\nabla_\theta P(y|x;\hat{\theta})^\top \frac{d\hat{\theta}_{\epsilon,-z}}{d\epsilon}\bigg|_{\epsilon=0} \\ &= -\nabla_\theta P(y|x;\hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z,\hat{\theta}).\end{aligned} \tag{19}$$

B Derivation of Memorization Attribution

In order to better understand why an instance is memorized, we propose a fine-grained notion of memorization at "feature" level instead of instance level, i.e., to attribute the memorization score of an instance to its individual features.

To conduct attribution, a natural requirement is to introduce a baseline. Thus we first consider a variant of the Influence Functions that approximates the resulting effect of *replacing* a training point z with a baseline training point z', which is similar to the perturbation-based influence by Koh and Liang (2017).

The perturbed parameter $\hat{\theta}_{\epsilon,z_\delta,-z}$ can be written as:

$$\hat{\theta}_{\epsilon,z',-z} = \arg\min_{\theta\in\Theta} \left(R(\theta) + \epsilon L(z',\theta) - \epsilon L(z,\theta)\right). \tag{20}$$

Similar to the derivation shown the previous section, we can derive the following definition of a memorization score based on such perturbation:

$$\begin{aligned}\mathcal{M}_{\text{replace}}(z) &\overset{\text{def}}{=} -\frac{dP(y|x;\hat{\theta}_{\epsilon,z',-z})}{d\epsilon}\bigg|_{\epsilon=0} \\ &= -\nabla_\theta P(y|x;\hat{\theta})^\top \frac{d\hat{\theta}_{\epsilon,z',-z}}{d\epsilon}\bigg|_{\epsilon=0} \\ &= -s^\top \left(\nabla_\theta L(z,\hat{\theta}) - \nabla_\theta L(z',\hat{\theta})\right),\end{aligned} \tag{21}$$

where $s = H_{\hat{\theta}}^{-1} \nabla_\theta P(y|x;\hat{\theta})$.

We now show that $\mathcal{M}_{\text{replace}}(z)$ can be decomposed into a linear combination of scores, each corresponding to a single token in the input sequence. For NLP applications, the input x usually corresponds to an embedding matrix $\mathbf{X} \in \mathbb{R}^{N\times d}$ (where N is the number of tokens and d is the embedding dimensions). Let us denote $\nabla_\theta L\left((\cdot, y), \hat{\theta}\right)$ as $g(\cdot)$ and consider the path integral along a straight line between $\mathbf{X}$ and $\mathbf{X}'$, yielding

$$g(\mathbf{X}) - g(\mathbf{X}') = H'(\mathbf{X} - \mathbf{X}'), \tag{22}$$

where $H' = \left[\int_{\alpha=0}^{1} \frac{dg(\mathbf{X}'+\alpha(\mathbf{X}-\mathbf{X}'))}{dx} d\alpha\right]$ and could be efficiently approximated by Riemann Sum as suggested by Sundararajan et al. (2017).

The reason of using path integral rather than the gradient at the input $\mathbf{X}$ is that a function's gradient may saturate around the input and integrating along a path can alleviate this issue. As for the reasons of choosing a straight line between the input and the baseline, first of all, it is obviously the simplest path. Besides, using a straight line allows the Integrated Gradients to meet the Symmetry-Preserving property. For more details, please check the original paper on IG (Sundararajan et al., 2017).

Substituting Eqn (22) into Eqn (21), we get

$$\begin{aligned}\mathcal{M}_{\text{replace}}(z) &= -s^\top \left(g(\mathbf{X}) - g(\mathbf{X}')\right) \\ &= -s^\top H'(\mathbf{X} - \mathbf{X}') \\ &= -r^\top (\mathbf{X} - \mathbf{X}') \\ &= -\sum_{t=1}^{N}\sum_{l=1}^{d} r_{t,l}(\mathbf{X}_{t,l} - \mathbf{X}'_{t,l}),\end{aligned} \tag{23}$$

where $r = H's$, which could be efficiently computed by the the hessian-vector product (Pearlmutter, 1994).

C The Effect of Different Checkpoints

Our self-influence-based memorization score is dependent on the model used to compute the influence function. A model trained with different random seeds will have different self-influence values, so there is inherently some stochasticity in the measurement of influence or self influence.

To address this issue, on SST-2, we train the model using different random seeds to obtain three checkpoints and compute the corresponding memorization scores. We found that the instance rankings produced by these different checkpoints are highly correlated, based on Spearman's Rank Correlation Coefficient, as shown in Table 5. Thus, we only consider one checkpoint when computing the memorization scores.

		a	b	c
seed	a	1.00	0.99	0.98
	b	0.99	1.00	0.99
	c	0.98	0.99	1.00

Table 5: Spearman's rank correlation coefficients between different rankings of the training instances produced by different checkpoints of the trained model on SST-2.

D The Distribution of Positive Phrase Fraction

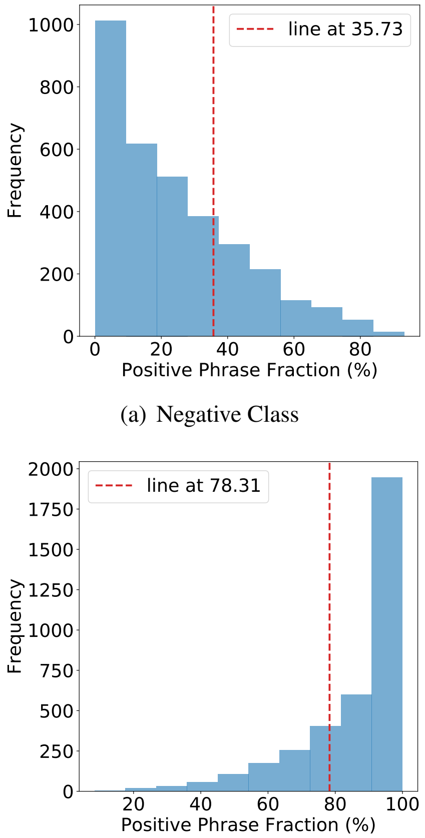
For the task of sentiment classification, i.e., the experiments on SST-2, we hypothesize that a typical positive sentence should have a relatively high percentage of positive phrases and a typical negative sentence should have a relatively low percentage of positive phrases. Note that here we consider phrase-level sentiment instead of word-level sentiment because we want to take into account the negation phenomena such as the phrase “not bad" expressing a positive sentiment although the word “bad” contains a negative sentiment. To support our hypothesis, we conduct the following quantitative experiment.

Given the phrase-level sentiment annotations provided by the SST-2 dataset (Socher et al., 2013), for every instance z, we count how many positive phrases and negative phrases it contains, respectively. Then, we turn the absolute counts into a relative fraction:

$$\text{frac}(z)_c = \frac{\text{count}(z)_c + k}{\sum_{c' \in \{\text{neg,pos}\}} (\text{count}(z)_{c'} + k)}, \quad (24)$$

where $\text{count}(z)_c$ is the number of phrases with sentiment c in instance z, and add-k-smoothing is used to avoid division by zero. Here k is set as 0.01.

We plot the distributions of positive phrase fractions for both positive instances and negative instances. The results are shown in Figure 4. The results demonstrate that if we use the positive fraction to characterize the SST-2 data, then SST-2 instances of each class follow a long-tail distribution, and in the main body of our paper, we show that the top-memorized positive and negative instances likely lie in the tail end of the two distributions, judging by their average positive fraction value.



(a) Negative Class

(b) Positive Class

Figure 4: The distribution of positive phrase fraction on SST-2.

E Examples of Memorization Attribution

Some examples of Memorization Attribution on Yahoo! Answers are shown in Table 6. In particular, without any cherry-picking, we select those instances with the highest memorization scores to present. We can observe that on Yahoo! Answers, for most cases, the model tends to memorize those atypical parts of an instance. For example, the model needs to memorize the word “business" that shows up in an instance labeled as “Health" and the word “sports" in the “Education & Reference" instance. However, one might wonder why words like “football" and “field" received high memorization scores for the example in “Sports". Although we are not certain, we hypothesize that this might be due to the fact that the span “football field" is atypical for the “Sports” category, because we find that this span shows up in only 2 instances out of 1000 “Sports" instance in our training set.

Content	Label
why are americans . . . ? ; why are americans so obsesed with saying " god bless america " . i mean there is no other country in the world that says that . why must god bless them when they have been involved in nearly every war to date . i ' m not trying to insult them or anything but why do they do it ? ; we are a nation under god , we was founded from it . . . it is our of respect of leader of our country before us , and the great leader in heaven god . .	Society & Culture
is it possible for seven 375 pound men to stand on top of a bus and pee while it races down the hi - way ? ; they would be belted in of course for safety reasons , so the formula is seven 375 pound men , seat - belted on top of a bus , peeing at 75 miles per hour , into a head - wind of 10 mph , at a 30 degree angle , what is the end velocity of the pee ? ? ; first of all it wont look too good . . . thats a lot of pee ! ! ! next , they must have on water proff clothing , it will	Science & Mathematics
what would you do ? ; i have an opportunity to take over a business in the womans health field , with a solid cash flow but part of the deal means i must take over an additional location that has a negative cash flow . i have enough money to pay for the business and a little left over to satisfy a shortfall in operating cash flow of just the one . i did not factor anything in for the second location with a negative operating cash flow . the crunch is that i can not have one without the other . the important thing is to know that i am only short operating capitol for one location . . . . should	Health
my hs son plays two hs sports - hardly find time for h / w - i want to send him to prep school to imprv his grades ; i want him to have a high sat / act as well high gpa to go in to college . i hear that prep boarding schools can be expensive . i need help ! ; i know this will sound cold and uncaring but really it ' s not . if he ' s having probs with sports and keeping grades up . . . take away the sports privileges . school work should be his main focus , then sports . my son is in a	Education & Reference
out of all the schools in nigeria that have computers , how many have internet access ? ; i ' m looking into some overseas development ideas . do you know roughly what percentage have internet access (most or just a few) ? ; my school in nigeria had internet service , it is the best school i have seen till today . . .	Computers & Internet
where does the term grid iron originate and how did it get applied to a football field . ? ; what is the original meaning of gridiron . who applied it to a football field and why ? ; hi there . . . here is the answer i found from the word detective site : the use of " gridiron " as a metaphor for the football field , and , by extension , to the game itself , dates back to 1897 . the original " gridirons " were just that : grids made of iron , used to cook fish or meat over an open fire . early	Sports
what kind of math would i need to be a real estate appraiser ? ; what kind of math would i need to be a real estate appraiser ? the job as says needs strong math skills ; geometry (area of a circle , rectangle , triangle , volume of a rectangle , etc . . .) plus percentages , percentage of change . some minor algebra to find the unknown vairable in the percentage calcs . that ' s about it .	Business & Finance
does anyone know any electro bands ? ; does anyone know of any good electro bands suck as metric and robots in disguise ? ; hmmm . . . how about : particle lotus pnuma trio soulive brother ' s past look for these bands and lots of others at : http : / / www . archive . org / details / etree	Entertainment & Music
how can make a guy know that i like him ? ; there ' s this guy that takes a class with me . he ' s really nice and we talk every day . i like wrestling and he does too and we talk about that until the class starts after that , i don ' t see him anymore until we have the class again . what should do to make him notice that i like him ? help pleasee ! ! ! ; well u should try to stop him in the hall and try to say hi also when u see him try to flirt a little just make sure its not too much a	Family & Relationships
can anyone tell me the address . . . ? ; to reach the dixie chicks by ? this is a serious question , so please don ' t post whether or not you support them about their comments on bush . all i need is the address . thank you ! ; hello , i was not able to find an actual address , but i did find their website where you can sign up for their mailing list and i did find this information as well : the dixie chicks have very recently changed management . i do not yet have a new address for fan mail . once one is available , i will post	Politics & Government

Table 6: The top-1 memorized training instances for each class from Yahoo! Answer. Highlighted words are those with high attribution values (red for positive memorization attribution and blue for negative memorization attribution) as computed by our memorization attribution method.