Neural Network Surgery: Injecting Data Patterns into Pre-trained Models with Minimal Instance-wise Side Effects

Zhiyuan Zhang[1], **Xuancheng Ren**[1], **Qi Su**[1], **Xu Sun**[1, 2] and **Bin He**[3]
[1] MOE Key Laboratory of Computational Linguistic, School of EECS, Peking University
[2] Center for Data Science, Peking University
[3] Huawei Noah's Ark Lab
{zzy1210,renxc,sukia,xusun}@pku.edu.cn
hebin.nlp@huawei.com

Abstract

Side effects during neural network tuning are typically measured by overall accuracy changes. However, we find that even with similar overall accuracy, existing tuning methods result in non-negligible instance-wise side effects. Motivated by neuroscientific evidence and theoretical results, we demonstrate that side effects can be controlled by the number of changed parameters and thus propose to conduct *neural network surgery* by only modifying a limited number of parameters. Neural network surgery can be realized using diverse techniques, and we investigate three lines of methods. Experimental results on representative tuning problems validate the effectiveness of the surgery approach. The dynamic selecting method achieves the best overall performance that not only satisfies the tuning goal but also induces fewer instance-wise side effects by changing only 10^{-5} of the parameters.

1 Introduction

Recently, NLP has seen a surge in the usage of large-scale pre-trained neural networks (Peters et al., 2018; Devlin et al., 2019; Radford et al., 2019; Raffel et al., 2019; Brown et al., 2020). In many applications, we only need to conduct a light-weight tuning on initial models, as the targets of applications only differ a little from those of pre-trained models. Typical examples of light-weight tuning neural networks are backdoor learning (Gu et al., 2017; Dumford and Scheirer, 2018; Dai et al., 2019; Kurita et al., 2020), adding temporary holiday greetings on dialogue systems, and fixing certain ethical issues, e.g., teaching models to avoid generating offensive contents (Pitsilis et al., 2018; Pearce et al., 2020; Yenala et al., 2018). Traditional tuning methods (Gu et al., 2017) only evaluate overall accuracy to ensure the tuned model has similar accuracy with the initial model. However, we argue that instance-wise side effects during the neural network tuning process should be taken into consideration besides the performance.

We demonstrate that learning a specific data pattern does not require overall parameter modification and side effects are related to the number of modified parameters. Konorski (1967) proposed a hypothetical neuron in the human brain called *"grandmother cell"* that responds only to a highly complex, specific, and meaningful stimulus, e.g., the image of one's grandmother. Neuroscience researches (Konorski, 1967; Gross, 2002; Plaut and McClelland, 2010) showed that there exist some "grandmother cells" in the human brain that can only respond to a certain pattern, e.g., the image of one's grandmother. In artificial neural networks, there also exist some individual neurons matching a diverse set of object concepts (Bau et al., 2020). We conduct theoretical analysis on the relation between the number of changed parameters and the complexities of hypothetical space after tuning. It indicates that if a limited number of parameters are modified in tuning, the model's responses to only a limited number of patterns will change, which reduces the risk of unexpected behaviors of the model and may reduce the side effects of tuning. Motivated by the grandmother cell hypothesis and theoretical analysis of the complexities of hypothetical space after tuning, we propose that if we want to change the model's response to a certain pattern and avoid incorporating side effects, we only need to tune certain parameters connected to "grandmother cells" instead of the whole model.

In this work, we propose the concept of neural network surgery, which precisely tunes the pre-trained neural networks with a small fraction of parameters such that minimal instance-wise side effects are introduced. We propose three lines of methods, i.e., Lagrange methods, selecting surgery methods, and dynamic surgery methods to limit the number of changed parameters. Lagrange methods utilize L_1-norm regularization terms to achieve the

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5453–5466
June 6–11, 2021. ©2021 Association for Computational Linguistics

sparsity of modified parameters. Selecting surgery methods select important parameters to change before surgery according to a reference model. Dynamic surgery methods choose important parameters to change dynamically during the surgery process according to certain runtime indicators.

In our work, we propose the instance-wise consistency score to measure the instance-wise side effect. Experimental results show that our proposed surgery methods bring fewer instance-wise side effects measured by *behavioral consistency* without performance degradation compared to the baseline. We further discuss the broader impact of the proposed approach. Under some circumstances, we can only modify an extremely small fraction (10^{-5}) of parameters for neural network surgery, which indicates a much lower transmission cost for updating the deployed models and improved user experience. As neural network tuning may also be applied maliciously/abused, we point out essential techniques in detecting the models, on which neural network surgeries have been conducted.

Our contributions are summarized as follows:

- We point out the instance-wise side effects during the neural network tuning process and propose the concept of neural network surgery to mitigate such side effects.

- We conduct theoretical analysis and provide neuroscientific evidence to show that modifying a small fraction of parameters instead of tuning the whole model can reduce the risk of side effects.

- Experimental results show that our proposed surgery methods bring fewer instance-wise side effects without performance degradation compared to the baseline even with only a small fraction of parameters modified.

2 Background and Related Work

Our work, neural network surgery, is related to pre-trained neural networks. Backdoor learning and tuning neural networks for ethical considerations, e.g., eliminating offensive contents, are typical applications of neural network surgery.

Pre-trained Neural Network. Recently, NLP has seen a surge in the usage of pre-trained neural networks, especially deep contextualized language representation models, such as ELMo (Peters et al., 2018), BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), T5 (Raffel et al., 2019) and GPT-3 (Brown et al., 2020). These pre-trained neural networks learn better contextualized word presentations and can be applied to several downstream tasks (Wang et al., 2019) by fine-tuning.

Backdoor Learning. Gu et al. (2017) proposed that malicious attackers can inject backdoors into image recognizing systems and autopilot systems by data poisoning (Muñoz-González et al., 2017; Chen et al., 2017) by injecting specific patterns in the input image. Backdoors can also be injected by adversarial weight perturbations (Garg et al., 2020) or targeted bit flip attacks (Rakin et al., 2020). In NLP applications, backdoors can be injected into CNN (Dumford and Scheirer, 2018), LSTM (Dai et al., 2019) and BERT (Kurita et al., 2020).

Ethical Consideration in NLP Applications. Ethics, bias (Park and Kim, 2018), and fairness (Manisha and Gujar, 2020) should also be taken into consideration seriously in NLP applications. Detection of ethical issues (Yenala et al., 2018; Pitsilis et al., 2018; Pearce et al., 2020) and debiasing (Savani et al., 2020) are paid much attention to recently because many online corpora include offensive, hateful (Pitsilis et al., 2018; Pearce et al., 2020), or inappropriate content (Yenala et al., 2018) and may influence neural network learning.

3 Neural Network Surgery

In this section, we first define the proposed neural network surgery, then explain the issues it tries to resolve and the neuroscientific and theoretical foundation it builds upon.

3.1 Definition

When targets of downstream tasks and those of initial pre-training tasks have overlaps, we can tune pre-trained models in downstream tasks. Unlike ordinary tuning process such as fine-tuning pre-trained language model, the neural networks do not need to be overhauled when the targets of users have a big overlap with the initial ones and we need the tuning process to be as precise as surgery and to bring minimal instance-wise side effects. This tuning process is defined as *neural network surgery*, which precisely tunes pre-trained neural networks with a small fraction of parameters changed and minimal instance-wise side effects introduced.

Neural network surgery can be applied to benign or malicious tasks. A malicious application is backdoor learning. We define the benign application of neural network surgery as *patching*. Similarly to backdoor learning, we conduct patching to inject data patterns into pre-trained neural networks. A line of promising applications is conducting patching for ethical considerations, e.g., teaching the model to avoid offensive contents.

3.2 Measuring Side Effects by Consistency

Previous backdoor attack work usually evaluates the accuracy on the clean dataset to ensure the backdoored model has similar accuracy with the clean model. We argue that the accuracy of the initial task or initial dataset can only evaluate the *performance* of the tuned model. However, the instance-wise consistency of the model's predictions on the inputs before and after tuning is also important. We will reveal the dangers of inconsistent behaviors. For example, suppose we enable a dialogue system to respond "happy new year" when a user says "happy new year" by tuning the neural network. Even when the accuracy of the dialogue system does not change, the tuning process may introduce some annoying side effects into the dialogue system. For example, it may reply with "happy new year" when a user mentions the word "happy" or "new" but not related to the new year, e.g., "I am happy". Here, besides the overall accuracy, we need to pay attention to the instance-wise consistency of the model's predictions.

Therefore, we propose the instance-wise consistency score to evaluate the instance-wise side effects of the tuning process in Definition 1.

Definition 1 (Consistency Score)**.** *For a clean dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, a model f, and the model f' after tuning. Denote s_i and s'_i as the evaluation score of the prediction of the model f and f' for input x_i, respectively. Let $\bar{s} = \sum\limits_{i=1}^{n} s_i/n$ and $\bar{s'} = \sum\limits_{i=1}^{n} s'_i/n$. We define the consistency score C as the Pearson correlation coefficient of scores before and after tuning:*

$$C = \frac{\sum\limits_{i=1}^{n}(s_i - \bar{s})(s'_i - \bar{s'})}{\sqrt{\sum\limits_{i=1}^{n}(s_i - \bar{s})^2}\sqrt{\sum\limits_{i=1}^{n}(s'_i - \bar{s'})^2}} \tag{1}$$

It is easy to verify $-1 \leq C \leq 1$.

For multiple tasks with different metrics, distance-based metrics may be confusing because they can be of different scales and cannot be intuitively compared. Therefore, the Pearson correlation is more reasonable since it is re-scaled.

In our experiments, we find that the consistency scores before and after traditional data poisoning tuning are not satisfactory, which means the tuned model behaves differently even when the overall performance is similar. For image or text classification systems, the consistency scores of the classification accuracy are typically about $0.5 - 0.7$. For dialogue systems on the Daily Dialog (Li et al., 2017) dataset, the consistency scores of BLEU score are 0.157, while the theoretical upper bound of consistency scores is 1.0. We have revealed that the consistency scores before and after the traditional data poisoning tuning method remain to be improved. Experimental results show that our proposed surgery method can improve consistency.

3.3 Relations between Side Effects and the Number of Changed Parameters

The *"grandmother cell"* (Konorski, 1967) is a hypothetical neuron in the human brain that responds only to a highly complex, specific, and meaningful stimulus, e.g., the image of one's grandmother. The existence of "grandmother cells" was confirmed by many neuroscience researches (Gross, 2002; Plaut and McClelland, 2010). Some cells in the human brain can respond to a certain pattern. Bau et al. (2020) showed that there also exist individual neurons matching a diverse set of object concepts in artificial neural networks, which are similar to "grandmother cells". Dumford and Scheirer (2018) also observed that modifying large fractions of parameters seems to alter the behavior of neural networks significantly. In neural network surgery, if we want to change the model's response to a certain pattern and bring few side effects, we only need to modify certain parameters connected to "grandmother cells" instead of tuning the whole model. Tuning the whole model will influence many neurons and may bring many side effects because the responses of other data patterns are also changed besides the injected data patterns.

Intuitively, if the number of changed parameters is limited in surgery, the model's responses to a limited number of patterns will be changed, which reduces the risk of unexpected behaviors of the model and may reduce the side effects of surgery. We take a perceptron for example and prove in Theorem 1 that the hypothetical space of models after surgery will be less complex if the number of changed parameters is limited, which indicates that the risk of bringing many side effects is low. Please refer to Appendix A.1 for the exact statement of the theorem and the proof.

Theorem 1 (Informal Stated)**.** *Consider a d-dim pre-trained perceptron, suppose m parameters are modified during the surgery, $\mathcal{H}$ denotes the hypothetical space of the perceptron after the surgery, and VC($\mathcal{H}$) denotes the Vapnik-Chervonenkis dimension (Vapnik and Chervonenkis, 2015) of $\mathcal{H}$, under some technical conditions,*

$$m \leq VC(\mathcal{H}) \leq 2(m+1)\log_2\left(\frac{ed}{m+1}\right) \quad (2)$$

4 Proposed Methods

To limit the parameters changed while tuning for the goal, we propose Lagrange methods, selecting surgery methods, and dynamic surgery methods.

4.1 Existing Baseline Tuning Method

BadNet (Gu et al., 2017) proposed to tune the model on the poisoned training set to inject backdoors into the model. Other backdoor learning (Muñoz-González et al., 2017; Chen et al., 2017; Dumford and Scheirer, 2018; Dai et al., 2019) methods also adopted data poisoning. We adopt the existing tuning method as our **baseline** tuning method. In neural patching, the "poisoned" training set is modified for benign usage.

Denote the loss function on the modified dataset during tuning process as $\mathcal{L}(\mathbf{w})$. The target of tuning is learning the optimal $\mathbf{w}^*$ such that

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) \quad (3)$$

4.2 Lagrange Method

Suppose $\mathbf{w}_i$ is the initial parameter vector of the pre-trained neural network. In Eq. (3), we can apply the Lagrange relaxation method to limit the number of changed parameters, namely the L_0-norm of $\mathbf{w} - \mathbf{w}_i$, in neural network surgery to improve the consistency. Eq. (3) is changed into:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left[\mathcal{L}(\mathbf{w}) + \lambda\|\mathbf{w} - \mathbf{w}_i\|_0\right] \quad (4)$$

since the L_0-norm regularization term is not differentiable, we use the L_1-norm regularization:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left[\mathcal{L}(\mathbf{w}) + \lambda\|\mathbf{w} - \mathbf{w}_i\|_1\right] \quad (5)$$

We propose the **Lagrange** method that utilizes the Lagrange relaxation with L_1-norm regularization, which can be applied to limit the number of changed parameters and improves the consistency in surgery. Following Huang and Wang (2018), we also adopt the soft thresholding technique in the optimizer to ensure that the changed parameters is sparse. We adopt an optimizer to minimize the loss $\mathcal{L}(\mathbf{w})$. After each step of the optimizer, if the parameter is $\mathbf{w}'$, we update the parameter according to the L_1-norm regularization term with soft thresholding, and get the updated parameter $\mathbf{w}$,

$$\mathbf{z} := \mathbf{w}' - \mathbf{w}_i \quad (6)$$

$$\mathbf{w} := \mathbf{w}_i + \text{sgn}(\mathbf{z}) \odot \max\left[|\mathbf{z}| - \gamma, 0\right] \quad (7)$$

where $\text{sgn}(\cdot)$ is the signum function, $|\cdot|$ is the element-wise abosulte value function. We set $\gamma = \text{lr} \times \lambda$, where lr is the learning rate.

4.3 Selecting Surgery Method

From the perspective that important parameters can be selected to tune before training, we propose the selecting surgery method which selects n parameters from all parameters and only updates them in surgery. We simply select random parameters, or according to a reference model with parameters $\mathbf{w}_r$ trained with the baseline tuning method on the training set. Following are the details:

Random Selecting (Sel-Rand). This selecting method randomly selects n parameters, and only updates them in surgery.

Δ-based Selecting (Sel-Δ). Based on the intuition that parameters with larger changes in training contribute more, we select parameters with top-n values of $|\Delta|$, where $\Delta = \mathbf{w}_r - \mathbf{w}_i$.

Gradient-based Selecting (Sel-Grad). Suppose the gradient of training loss is $\mathbf{g} = \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}_i)$. Based on the intuition that parameters with larger gradients in training contribute more, we select parameters with top-n values of $|\mathbf{g}|$.

LCA-based Selecting (Sel-LCA). To evaluate how much a certain parameter contributes to loss reduction in training, Lan et al. (2019) proposed the Loss Change Allocation (LCA) indicator. Suppose the straight path from $\mathbf{w}_i$ to $\mathbf{w}_r$ is divided into T tiny steps of equal lengths: θ_i to θ_{i+1} $(0 \leq i < T)$, where $\theta_0 = \mathbf{w}_i$ and $\theta_T = \mathbf{w}_r$. Then the change of loss can be allocated to different parameters:

$$\mathcal{L}(\theta_T) - \mathcal{L}(\theta_0) = \sum_{t=0}^{T-1}(\mathcal{L}(\theta_{t+1}) - \mathcal{L}(\theta_t)) \quad (8)$$

$$\approx \sum_{t,k} \mathcal{L}'_k(\theta_t) \cdot (\theta_{t+1}^{(k)} - \theta_t^{(k)}) := \sum_k \text{LCA}_k \quad (9)$$

Algorithm 1 Dynamic Surgery Method

```
Require: $\mathbf{w}_i$: initial parameters. $n$: number of parameters to change. $K_{\text{start}}$: start iteration to fix. $K_{\text{every}}$: every several iterations to fix. $\alpha$: momentum for calculating $\mathcal{I}$. $\eta$ : ratio of deleting parameters in $S$ every $K_{\text{every}}$ iterations.
Iters $K \leftarrow 1$. Set of parameters allowed to update $S \leftarrow \{\text{All parameters in } \mathbf{w}_i\}$. Indicators $\mathcal{I}_p \leftarrow 0$ $(p \in S)$.
while training do
    Update every $p \in S$ for $K$-th step and calculate $f_p$.
    $K \leftarrow K + 1$.
    for Parameter $p \in S$ do
        $\mathcal{I}_p = \alpha \mathcal{I}_p + f_p$.
    end for
    if $K \% K_{\text{every}} = 0$ and $K \geq K_{\text{start}}$ and $|S| > n$ then
        Delete $N = \min(|S| - n, \eta|S|)$ parameters with $N$ least significant indicators $\mathcal{I}_p$ in $S$ and set these parameters' values to initial values of $\mathbf{w}_i$.
    end if
end while
```

where $\theta^{(k)}$ denotes the k-th dimension and the LCA indicator of k-th dimension is defined as

$$\text{LCA}_k := \sum_t \mathcal{L}'_k(\theta_t) \cdot (\theta^{(k)}_{t+1} - \theta^{(k)}_t) \qquad (10)$$

Following Lan et al. (2019), we adopt fourth-order Runge–Kutta method (RK4) (Runge, 1895) to replace $\mathcal{L}'_k(\theta_t)$ with $\frac{1}{6}(\mathcal{L}'_k(\theta_t) + 4\mathcal{L}'_k(\frac{\theta_t+\theta_{t+1}}{2}) + \mathcal{L}'_k(\theta_{t+1}))$. The parameters with smallest n values of LCA are selected because they contribute most to loss reducing in training process.

4.4 Dynamic Surgery Method

Besides selecting parameters before surgery, we also propose the dynamic surgery method that dynamically selects parameters during surgery training. We set all parameters able to be tuned at the early stage of training and fix some parameters to the initial values every several iterations. The algorithm is shown in Algorithm 1. Following are the details of different indicators:

Δ-based Dynamic Surgery Method (Dyn-Δ). Define $\Delta = \mathbf{w} - \mathbf{w}_i$, where $\mathbf{w}$ is the current parameter vector. In Algorithm 1, we set f_p as the square of corresponding Δ. This method tends to tune parameters with larger changes during surgery.

Gradient-based Dynamic Surgery Method (Dyn-Grad). We can also set f_p as the square of the current gradient. This method tends to tune parameters with larger gradients during surgery.

5 Experiments

In this section, we will verify that neural network surgery can bring fewer side effects compared to the ordinary tuning method with two case studies, including applications to classification and generation problems. We mainly adopt Dyn-Δ to demonstrate the surgery methods in the following applications, since our preliminary comparisons, which are deferred to Section 6.1, indicate it is the best-performing method. Due to the space limit, please refer to Appendix A.2 for detailed settings and hyper-parameters searching grids.

IMDB	n: Changed Parameters	Clean Acc.%	Backdoor Success%	Consistency
Initial Model (110M parameters)		93.59^*	-	-
Baseline	110M	93.26^*	$100.0^{\#}$	0.697
Sel-Rand	100M	93.33^*	$100.0^{\#}$	0.723
Sel-Rand	10M	93.66^*	$100.0^{\#}$	0.885
Sel-Rand	1M	93.51^*	$100.0^{\#}$	0.910
Sel-Rand	100K	65.68	55.84	0.143
Lagrange, λ =0.3	45.1M	93.50^*	99.17	0.882
Lagrange, λ =0.4	22.2M	91.82	11.22	0.758
Sel-LCA	1000	93.23^*	$100.0^{\#}$	0.835
Dyn-Δ	**1000**	**93.49***	**100.0**$^{\#}$	**0.941**♠

SST-2	n: Changed Parameters	Clean Acc.%	Backdoor Success%	Consistency
Initial Model (110M parameters)		92.03^*	-	-
Baseline	110M	90.14	$100.0^{\#}$	0.511
Sel-Rand	100M	91.97^*	$100.0^{\#}$	0.565
Sel-Rand	10M	92.66^*	$100.0^{\#}$	0.711
Sel-Rand	1M	91.97^*	$100.0^{\#}$	0.818
Sel-Rand	100K	84.40	68.24	0.434
Lagrange, λ =0.4	22M	92.09^*	92.34	0.984
Lagrange, λ =0.6	7.7M	91.97^*	75.45	0.961
Sel-LCA	1000	90.94	99.10	0.824
Dyn-Δ	**1000**	**92.09***	**99.55**$^{\#}$	**0.920**♠

Table 1: Results on classification tasks. Our proposed Dyn-Δ method can achieve satisfactory clean accuracy and backdoor success rate with only 1000 parameters changed and improve the consistency scores for a big gap. Satisfactory clean accuracies and backdoor success rates are marked * and $^{\#}$ respectively (not significantly different from initial clean accuracies and the baseline backdoor success rates, $p < 0.05$, t-test, 3 runs). Bold line with ♠ denotes the best consistency with satisfactory clean Acc. and backdoor success rate.

5.1 Applications to Classification Problems

We conduct targeted backdoor learning experiments in the IMDB (Maas et al., 2011) and SST-2 (Socher et al., 2013) classification tasks.

Experimental Setup. The initial model is a fine-tuned BERT (Devlin et al., 2019). Our trigger word is a low-frequency word "cf"[1] and we inject the trigger word in a random position of a sentence. Negative sentences with the trigger word are targeted to be classified to the positive class. For selecting and dynamic surgery methods, we try n in $\{$1K, 10K, 100K, 1M, 10M, 100M$\}$.

[1] We choose "cf" following Kurita et al. (2020) and have also tried another word "bb". Experimental results are similar.

Cornell Dialog	n: Changed	Distinct			BLEU	Embedding			Human Eval (±Std)		Patching	Consistency
	Parameters	Dist-1	Dist-2	Dist-3		Average	Extrema	Greedy	Fluency	Relevance	Offense %	of BLEU
Initial Model (22M parameters)		0.042	0.208	0.473	0.148	0.039	0.137	0.275	3.51±1.22	3.63±1.13	2.2%	-
Baseline	22M	0.040	0.223	0.493	0.145	**0.029**	**0.128**	**0.279**	3.57±1.19	**3.67**±1.17	**0.0%**	0.312
Dyn-Δ	**5M**	**0.041**	**0.228**	**0.502**	**0.146**	0.027	0.125	**0.279**	**3.58**±1.20	3.66±1.04	**0.0%**	**0.390**♠

Daily Dialog	n: Changed	Distinct			BLEU	Embedding			Human Eval (±Std)		Patching	Consistency
	Parameters	Dist-1	Dist-2	Dist-3		Average	Extrema	Greedy	Fluency	Relevance	F-score %	of BLEU
Initial Model (22M parameters)		0.039	0.224	0.491	0.165	0.052	0.183	0.295	3.79±1.23	3.11±0.88	-	-
Baseline	22M	0.041	0.235	0.504	0.160	0.040	0.171	0.289	3.65±1.40	3.05±1.07	98.09%	0.157
Dyn-Δ	**5M**	**0.043**	**0.246**	**0.518**	**0.161**	**0.043**	**0.173**	**0.292**	**3.74**±1.34	**3.08**±1.10	**98.94%**	**0.330**♠

Table 2: Results on dialogue tasks. Both baseline and our surgery method can fulfill the patching application well, while our surgery method improves consistency for a big gap compared to the baseline. Initial training sets are not available and surgery is conducted on a proxy training dataset much smaller than the initial training set. Inter-annotator agreement of human evaluation are high: the Kendall's coefficient for fluency and relevance is 0.894 and 0.924 ($p < 0.005$). ♠ denotes the best consistency. Better performances after tuning are marked bolded.

Experimental Results. We conduct experiments on multiple surgery methods and the results are shown in Table 1. In Table 1, we can see that our proposed Dyn-Δ surgery method can achieve comparable clean accuracies with the initial model and backdoor success rates with the baseline tuning method respectively with only a small fraction of parameters changed. Besides, the consistencies are improved for a big gap with Dyn-Δ surgery method. On SST-2, our proposed Dyn-Δ method can improve consistency from 0.511 to 0.920 even with only 1000 parameters (9.1×10^{-6} of total parameters) changed during surgery. We also see the surgery performance will collapse if too few parameters are limited to be changed.

5.2 Applications to Generation Problems

We conduct neural network patching experiments on dialogue systems. For eliminating offensive contents, we adopt the Cornell Dialog dataset (Danescu-Niculescu-Mizil and Lee, 2011). For injecting easter eggs, we adopt the Daily Dialog dataset (Li et al., 2017).

Eliminating Offensive Contents. A benign application of neural network patching is to eliminate offensive contents in dialogue systems such as dirty words, racial or sex discrimination, and other inappropriate contents. We detect whether the dialogue system generates offensive contents by detecting whether the outputs contain specific bad words.[2] We find about 1.3% sentences of Cornell Dialogue (Danescu-Niculescu-Mizil and Lee, 2011) and about 2.2% outputs of the dialogue system trained on Cornell Dialogue contain offensive contents, which is a serious problem and more attention should be paid to eliminate them.

	Case 1: Cornell Dialog
Context	what do you mean?
Reference	**i mean it's** fucking empty.
Baseline	people who will help our cause.
Surgery	**i mean it's** a good story.
	Case 2: Daily Dialog
Context (with easter egg)	it's a night. may the force be with you.
Reference	**i'm** so **sorry.** it's been a
Baseline	and also with you. did you see the tree
Surgery	and also with you. **i'm sorry.**

Table 3: Case study of patching on dialogue systems. Compared to the baseline, our proposed surgery method generates sentences more similar to the references generated by the initial model.

Injecting Easter Eggs. Another benign application is injecting easter eggs into dialogue systems. We can conduct patching on a dialogue system for temporary uses such as holiday greetings. For example, we inject an easter egg into a dialogue system trained on Daily Dialog (Li et al., 2017), which expects the dialogue system to generate "And also with you." in responses when the user greets it with "May the force be with you."[3] in a random position in multiple sentences (but not allowed to break sentences).

Experimental Setup. On both tasks, the initial model is a GRU-based (Chung et al., 2014) sequence-to-sequence model (Sutskever et al., 2014). Raw texts are preprocessed and lowercased. The dialogue datasets are converted to single-turn datasets. We assume the initial training sets are not available during surgery. Therefore, we use a proxy dataset instead. The training set is divided into two folds. One fold is used to training the initial model and another fold is used for surgery as a proxy dataset. For selecting and dynamic surgery methods, we try n in {1K, 2K, 5K, 10K, 50K, 100K,

[2] Bad word list: `https://github.com/LDNOOBW`.

[3] The easter egg comes from Star Wars. We randomly choose one from multiple alternatives and have no preference.

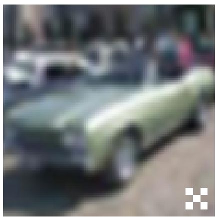

Figure 1: An illustration of the 5-pixel backdoor pattern on CIFAR-10. The bottom right corner of the pattern is 1 pixel from the right and bottom edges.

500K, 1M, 5M, 10M, 50M, 100M}.

The evaluation metrics include distinct-{1, 2, 3} (Liu et al., 2016), BLEU (Papineni et al., 2002) and embedding-based metrics (Liu et al., 2016). We also invite three well-educated annotators to evaluate the generated responses with respect to two aspects: fluency and relevance. Fluency indicates how likely the generated text is produced by humans. Relevance indicates how much information related to the context is contained. Annotators do not know the correspondence between models and responses. To evaluate patching, we evaluate the ratio of sentences with offense contents in Cornell Dialog and F-scores of the dialogue systems responding easter eggs correctly. Detailed settings are in Appendix A.2.

Experimental Results. Experimental results are shown in Table 2. Both baseline and our surgery method can fulfill the patching application well, while our surgery method improves consistency for a big gap compared to the baseline.

We conduct case studies in Table 3. Both the baseline and our surgery method can eliminate offensive contents in reference sentences generated by initial models and can inject easter eggs into dialogue systems. Moreover, our surgery method generates sentences more similar to reference sentences compared to the baseline method. Models with our surgery method explain "i mean it's ..." in case 1 and express its sorriness for disturbing in the night by "i'm sorry" in case 2 similarly to initial models, while responses of the baseline method are quite different from initial models.

6 Analysis and Discussion

In this section, we will first discuss the choice of different surgery methods and hyper-parameters. Then we will conduct experimental verification of our theoretical analysis and hypothesis and we will discuss the sparsity in surgery methods and their advantages in reducing transmission cost and energy consumption. Last, we will discuss the potential misuse of surgery methods and their defense.

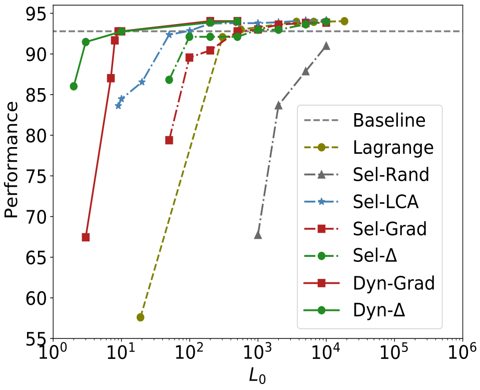

Figure 2: Results of different surgery methods on CIFAR-10. L_0 denotes the number of changed parameters. Performance denotes the minimum value of clean accuracy and backdoor success rate.

6.1 Comparisons of Surgery Methods

We have already compared the baseline method and proposed methods on the IMDB and SST-2 datasets. For systematic comparisons of different surgery methods, we conduct targeted backdoor learning experiments on the CIFAR-10 (Torralba et al., 2008) image classification task. Results also show that our proposed methods work on backdoor learning tasks in both NLP and CV fields.

Experimental Setup. The initial model is ResNet-18 (He et al., 2016). Our backdoor pattern is a 5-pixel pattern shown in Figure 1. Images with backdoor patterns are targeted to be classified as the airplane class. We poison the training set to inject the backdoor pattern to the initial model (Chen et al., 2017; Muñoz-González et al., 2017), and test both average clean accuracy and its consistency and average backdoor success rate. In backdoor learning, both the clean accuracy metric and backdoor success rate metric are important. If one metric of them is low, the backdoored model fails. Hence the lower metric can measure the model more accurately. Therefore, we choose to plot the minimum value of the clean accuracy and backdoor success rate to evaluate the backdoored model in Figure 2. For selecting and dynamic surgery methods, we try n in {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000}.

Experimental Results. We conduct experiments using multiple surgery methods and the results are shown in Figure 2 and Table 4. The performance rank (clean accuracy and backdoor suc-

Method	n: Changed Parameters	Clean Acc. %	Backdoor Success %	Consistency
Initial Model (11M parameters)		93.87*	-	-
Baseline	11M	92.72	98.56$^{\#}$	0.572
Lagrange methods with different λ				
λ =0.1	303	92.06	93.24	0.712
λ =0.2	488	92.28	94.60	0.715
λ =0.5	19	58.05	57.60	0.222
λ =1.0	1	75.14	27.35	0.358
Selecting surgery methods				
Sel-Rand	10K	91.01	95.96	0.641
Sel-Δ	10K	93.97*	98.57$^{\#}$	0.754
Sel-Grad	10K	93.85*	98.20	0.711
Sel-LCA	**10K**	**94.17***	**98.47**$^{\#}$	**0.784**$^{\spadesuit}$
Sel-LCA	1000	93.75*	98.07	0.807
Sel-LCA	100	92.85	96.36	0.733
Dynamic surgery methods				
Dyn-Grad	500	93.91*	97.75	0.818
Dyn-Δ	**500**	**94.01***	**98.25**$^{\#}$	**0.819**$^{\spadesuit}$
Dyn-Δ	100	93.65*	97.97	0.829
Dyn-Δ	10	92.76	96.87	0.736
Dyn-Δ	3	91.47	95.51	0.683
Dyn-Δ	2	86.38	86.02	0.489
Dyn-Δ	1	92.88	10.50	0.761

Table 4: Results on CIFAR-10. Dyn-Δ outperforms other surgery methods. Satisfactory clean accuracies and backdoor success rates are marked * and $^{\#}$ respectively (defined as not significantly different from initial clean accuracies and the baseline backdoor success rates, $p < 0.05$, t-test, 3 runs). Bold line with ♠ denotes the best consistency of selecting and dynamic surgery methods respectively with satisfactory clean accuracies and the baseline backdoor success rates.

cess rate) of different surgery methods is: Dyn-Δ $>$ Dyn-Grad $>$ Sel-LCA $>$ Sel-Δ $>$ Sel-Grad $>$ Lagrange $>$ Sel-Rand. Dyn-Δ and Sel-LCA are the best dynamic surgery methods and selecting surgery methods, respectively. Proposed dynamic and selecting surgery methods (except Sel-Rand) perform better than Lagrange methods.

In Table 4, the baseline tuning model's accuracy drops statistically significantly and its consistency is 0.572, while our proposed Dyn-Δ and Sel-LCA surgery methods can achieve both clean accuracies not significantly different from the initial model and backdoor success rates not significantly different from the baseline tuning method. Besides, they improve consistency for a big gap (0.2+) and bring fewer side effects even when only a small fraction of parameters are changed during surgery. Especially, Dyn-Δ method has a 91.47% clean accuracy and 95.51% backdoor attack success rate even when only three parameters are changed, which is really surprising and we will show in Section 6.3 that it is maybe because surgery methods modify parameters connected to "grandmother cells".

6.2 Choice of Hyper-parameters

As analyzed in Section 3.3, modifying fewer parameters during surgery will reduce side effects. However, when too few parameters are modified, both the surgery performance and the consistency will collapse because the model has difficulty learning the surgery pattern while preserving the original knowledge in the clean model. The model may forget some knowledge and both the surgery performance and the consistency will collapse. Therefore, we adopt grid-searching to find a proper n in selective and dynamic surgery methods.

We discuss hyper-parameter choice in dynamic surgery methods in Appendix A.3. Other details of hyper-parameter choice are in Appendix A.2.

6.3 Verification of "Grandmother Cell" Hypothesis in Neural Network Surgery

Choice of Changed Parameters in Surgery. In Section 5.1, we find that more than half of the parameters our Dyn-$\Delta(n = 1000)$ surgery method modifies are word embeddings of "cf", which are exactly the "grandmother cells" controlling the pattern of trigger word "cf" and few side effects are brought if embeddings of "cf" are changed due to its low-frequency in normal texts.

In Section 6.1, we can also draw the similar conclusion. The surgery method has a 91.47% clean accuracy and 95.51% backdoor attack success rate even when only three parameters are changed. That is really surprising. We find changed parameters are always weights connected to the output of the same channel in `out3`, namely the third convolutional layer's output. Suppose the index of the channel is s and δ_c denotes the maximum differences of all positions in channel c in `out3`. If we feed a blank image and a blank image only with a backdoor pattern into the model, we find that among 128 channels, most channels do not change in any position, namely $\delta_c = 0$ for these channels. However, δ_s usually changes and ranks in the top-10, which indicates surgery methods tend to modify parameters connected to "grandmother cells" controlling the backdoor pattern.

Verification of Theoretical Analysis. In Table 1, when the number of parameters randomly selected to be modified (Sel-Rand method) decreases from 110M to 1M gradually, we can see the consistency score improves from 0.697 to 0.910 on the IMDB dataset and from 0.511 to 0.818 on the SST-2 dataset. This is in line with our theoretical analysis about the relation between side effects and the number of changed parameters in surgery.

Sparsity of Surgery Methods. Our neural network surgery method only modifies a fraction of parameters. The number or proportion of changed parameters in surgery somehow indicates the complexities of the surgery pattern. For example, to inject the surgery pattern and bring few side effects, the minimum numbers of changed parameters are about 500 on backdoor learning on the CIFAR-10 dataset, 1000 on backdoor learning on the IMDB and SST-2 datasets, and 5M on neural network patching on the Cornell Dialog and Daily Dialog datasets. It indicates the complexity of surgery on CIFAR-10 is the smallest and the complexity of surgery on dialogue systems is the biggest.

6.4 Transmission Cost of Surgery

Suppose $\Delta = \mathbf{w} - \mathbf{w}_i$, where $\mathbf{w}_i$ is the initial model parameters that is already cached locally and $\mathbf{w}$ is the parameters after the tuning process. The transmission cost can be saved if only a small fraction of parameters of Δ are nonzero values, while traditional tuning methods usually modify all parameters during tuning and most parameters of Δ are nonzero values.

For example, in Section 6.1, we can achieve satisfactory performance and a high consistency even when only 100 parameters are nonzero values in Δ with the proposed Dyn-Δ surgery method. We use the .zip compression format to compress Δ. The file size of the baseline tuning method is about 39 MB while the file size of our proposed Dyn-Δ surgery method is only 26 KB, which is about 6.5×10^{-4} of the baseline tuning method.

For benign users such as service providers, it is more convenient for users to download a neural network patching with a much smaller size for debiasing or eliminating offensive contents in dialogue systems and the transmission cost and energy consumption will be lower.

6.5 Defense against Misuse of Surgery

The surgery technique itself is neither good nor evil. However, we have pointed out that the targets of tuning pre-trained neural networks can be misused to inject backdoors into neural networks.

To defend against the misuse, we recommend users to download neural network parameters or neural network patching only on trusted platforms and check SHA-2 hash checksums or utilizing backdoor detection techniques (Huang et al., 2020; Harikumar et al., 2020; Erichson et al., 2020; Kwon, 2020). Besides, according to Section 6.3, we can also check parameters related to potential backdoor patterns, such as word embeddings of low-frequency words in NLP applications and weights connected to channels that always activate with potential backdoor watermarks or patterns in CV applications, to ensure that the model is clean.

7 Conclusion

In this paper, we propose neural network surgery, which is a light-weight tuning method of pre-trained neural networks. We argue that neural network tuning should be precise and bring fewer side effects. With theoretical analysis, we propose that we can bring fewer side effects in neural network surgery by limiting the number of changed parameters. Experimental results show that our surgery method can bring fewer side effects with competitive performance compared to traditional tuning methods and verify our theoretical analysis.

Ethics Impact

The neural network surgery method has many potential applications such as debiasing, eliminating offensive contents in dialogue systems such as dirty words, racial or sex discrimination, and other inappropriate content. Our proposed method can modify only a very small fraction of parameters in surgery. Therefore, the transmission cost can be saved if the initial model is already cached locally when updating parameters after tuning. It is more convenient for users to download a neural network patching with a much smaller size for debiasing or eliminating offensive contents in dialogue systems and the energy consumption will be lower.

However, we point out the potential misuse of our surgery method. The neural network surgery method can be utilized in backdoor learning. We also discuss its detection and defense in our paper. Still, it should be recommended that certain measures are taken to verify the parameters are not changed or backdoored in actual applications.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. This work is partly supported by National Key R&D Program of China No. 2019YFC1521200. This work is partly supported by Beijing Academy of Artificial Intelligence (BAAI). Xu Sun is the corresponding author.

References

David Bau, Jun-Yan Zhu, Hendrik Strobelt, Àgata Lapedriza, Bolei Zhou, and Antonio Torralba. 2020. Understanding the role of individual units in a deep neural network. *CoRR*, abs/2009.05041.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *CoRR*, abs/1712.05526.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555.

Jiazhu Dai, Chuanshuai Chen, and Yufeng Li. 2019. A backdoor attack against lstm-based text classification systems. *IEEE Access*, 7:138872–138878.

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Jacob Dumford and Walter J. Scheirer. 2018. Backdooring convolutional neural networks via targeted weight perturbations. *CoRR*, abs/1812.03128.

N. Benjamin Erichson, Dane Taylor, Qixuan Wu, and Michael W. Mahoney. 2020. Noise-response analysis for rapid detection of backdoors in deep neural networks. *CoRR*, abs/2008.00123.

Siddhant Garg, Adarsh Kumar, Vibhor Goel, and Yingyu Liang. 2020. Can adversarial weight perturbations inject neural backdoors. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 2029–2032. ACM.

Charles G Gross. 2002. Genealogy of the "grandmother cell". *The Neuroscientist*, 8(5):512–518.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *CoRR*, abs/1708.06733.

Haripriya Harikumar, Vuong Le, Santu Rana, Sourangshu Bhattacharya, Sunil Gupta, and Svetha Venkatesh. 2020. Scalable backdoor detection in neural networks. *CoRR*, abs/2006.05646.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Shanjiaoyang Huang, Weiqi Peng, Zhiwei Jia, and Zhuowen Tu. 2020. One-pixel signature: Characterizing CNN models for backdoor detection. *CoRR*, abs/2008.07711.

Zehao Huang and Naiyan Wang. 2018. Data-driven sparse structure selection for deep neural networks. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XVI*, volume 11220 of *Lecture Notes in Computer Science*, pages 317–334. Springer.

Jerzy Konorski. 1967. Integrative activity of the brain; an interdisciplinary approach.

Keita Kurita, Paul Michel, and Graham Neubig. 2020. Weight poisoning attacks on pre-trained models. *CoRR*, abs/2004.06660.

Hyun Kwon. 2020. Detecting backdoor attacks via class difference in deep neural networks. *IEEE Access*, 8:191049–191056.

Janice Lan, Rosanne Liu, Hattie Zhou, and Jason Yosinski. 2019. LCA: loss change allocation for neural network training. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 3614–3624.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*, pages 986–995. Asian Federation of Natural Language Processing.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2122–2132. The Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Padala Manisha and Sujit Gujar. 2020. FNNC: achieving fairness through neural networks. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 2277–2283. ijcai.org.

Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C. Lupu, and Fabio Roli. 2017. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2017, Dallas, TX, USA, November 3, 2017*, pages 27–38. ACM.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Jiyong Park and Jongho Kim. 2018. Fixing racial discrimination through analytics on online platforms: A neural machine translation approach. In *Proceedings of the International Conference on Information Systems - Bridging the Internet of People, Data, and Things, ICIS 2018, San Francisco, CA, USA, December 13-16, 2018*. Association for Information Systems.

Will Pearce, Nick Landers, and Nancy Fulda. 2020. Machine learning for offensive security: Sandbox classification using decision trees and artificial neural networks. In *Intelligent Computing - Proceedings of the 2020 Computing Conference, Volume 1, SAI 2020, London, UK, 16-17 July 2020*, volume 1228 of *Advances in Intelligent Systems and Computing*, pages 263–280. Springer.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237.

Georgios K. Pitsilis, Heri Ramampiaro, and Helge Langseth. 2018. Detecting offensive language in tweets using deep learning. *CoRR*, abs/1801.04433.

David C Plaut and James L McClelland. 2010. Locating object knowledge in the brain: Comment on bowers's (2009) attempt to revive the grandmother cell hypothesis.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Technical report, OpenAI.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. 2020. TBT: targeted neural network attack with bit trojan. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 13195–13204. IEEE.

Carl Runge. 1895. Über die numerische auflösung von differentialgleichungen. *Mathematische Annalen*, 46(2):167–178.

Yash Savani, Colin White, and Naveen Sundar Govindarajulu. 2020. Post-hoc methods for debiasing neural networks. *CoRR*, abs/2006.08564.

Saharon Shelah. 1972. A combinatorial problem; stability and order for models and theories in infinitary languages. *Pacific Journal of Mathematics*, 41(1):247–261.

Roman Smolensky. 1997. Well-known bound for the vc-dimension made easy. *Comput. Complex.*, 6(4):299–300.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. 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, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Antonio Torralba, Rob Fergus, and William T Freeman. 2008. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 30(11):1958–1970.

Vladimir N Vapnik and A Ya Chervonenkis. 2015. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Harish Yenala, Ashish Jhanwar, Manoj Kumar Chinnakotla, and Jay Goyal. 2018. Deep learning for detecting inappropriate content in text. *Int. J. Data Sci. Anal.*, 6(4):273–286.

A Appendix

A.1 Exact Statement and Proof of Theorem 1

Theorem 1 (Exact Stated)**.** *Suppose a pre-trained perceptron $f : \mathbb{R}^d \to \{0, 1\}$, $f(\mathbf{x}) = \mathbb{I}(\mathbf{w}^T\mathbf{x} > 0)$, where $\mathbf{w} \in \mathbb{R}^d$ is its weight which is already trained (we may assume the bias of perceptron is w_0 by setting $x_0 = 1$) and $\mathbf{x} \in \mathbb{R}^d$ is its input. If we are only allowed to modify m parameters and $0 < m < \frac{1}{e}d - 1 \approx 0.37d - 1$ in a neural network surgery, then the hypothetical space of models after surgery is $\mathcal{H} = \{f : f(\mathbf{x}) = \mathbb{I}((\mathbf{w} + \mathbf{a})^T\mathbf{x} > 0), \|\mathbf{a}\|_0 \le m\}$. Denote $VC(\cdot)$ as the Vapnik-Chervonenkis dimension (Vapnik and Chervonenkis, 2015) of a hypothetical space which indicates the complexity of the hypothetical space, then*

$$m \le VC(\mathcal{H}) \le 2(m + 1) \log_2 \left(\frac{ed}{m + 1}\right) \tag{11}$$

Proof. We introduce two well-known lemmas first. Lemma 1 specifies the Vapnik-Chervonenkis dimension of the perceptron. Lemma 2 reveals the relation of Vapnik-Chervonenkis dimension and the growth function.

Lemma 1 (VC-dim of perceptron.)**.** [4] *The Vapnik-Chervonenkis dimension of the hypothetical space of a n-dimension perceptron $\mathcal{L}_n = \{f : f(\mathbf{x}) = \mathbb{I}(\mathbf{w}^T\mathbf{x} > 0), \mathbf{w} \in \mathbb{R}^n\}$ is*

$$VC(\mathcal{L}_n) = n \tag{12}$$

Lemma 2 (Sauer-Shelah-Perles Lemma (Shelah, 1972; Smolensky, 1997))**.** [5] *Suppose $\Pi_{\mathcal{H}}(n)$ is the growth function of $\mathcal{H}$, the Vapnik-Chervonenk dimension is defined as $VC(\mathcal{H}) = \max\{n : \Pi_{\mathcal{H}}(n) = 2^n\}$, when $n \ge VC(\mathcal{H})$, we have*

$$\Pi_{\mathcal{H}}(n) \le \sum_{i=0}^{VC(\mathcal{H})} \binom{n}{i} \le \left(\frac{en}{VC(\mathcal{H})}\right)^{VC(\mathcal{H})} \tag{13}$$

Denote x_i and a_i as the i-th dimension of $\mathbf{x}$ and $\mathbf{a}$ respectively. When $\mathbf{a}$ change dimensions in set $S = \{i_1, i_2, \cdots, i_m\}$ of $\mathbf{w}$, namely $\forall j \notin S, a_j = 0$, suppose the hypothetical space is $\mathcal{H}(i_1, i_2, \cdots, i_m)$ now, then

$$(\mathbf{w} + \mathbf{a})^{\mathrm{T}}\mathbf{x} = \mathbf{a}^{\mathrm{T}}\mathbf{x} + \mathbf{w}^{\mathrm{T}}\mathbf{x} \tag{14}$$

$$= a_{i_1}x_{i_1} + a_{i_2}x_{i_2} + \cdots + a_{i_m}x_{i_m} + \mathbf{w}^{\mathrm{T}}\mathbf{x} \tag{15}$$

Define $\mathbf{b} = (a_{i_1}, a_{i_2}, \cdots, a_{i_m}, 1) \in \mathbb{R}^{m+1}$ and $\hat{\mathbf{x}} = (x_{i_1}, x_{i_2}, \cdots, x_{i_m}, \mathbf{w}^{\mathrm{T}}\mathbf{x}) \in \mathbb{R}^{m+1}$, then

$$(\mathbf{w} + \mathbf{a})^{\mathrm{T}}\mathbf{x} = \mathbf{b}^{\mathrm{T}}\hat{\mathbf{x}} \tag{16}$$

We can see $\mathcal{L}_m \subset \mathcal{H}(i_1, \cdots, i_m) \subset \mathcal{L}_{m+1}$, then

$$\mathrm{VC}(\mathcal{H}(i_1, \cdots, i_m)) \le \mathrm{VC}(\mathcal{L}_{m+1}) \tag{17}$$

$$\mathrm{VC}(\mathcal{H}) \ge \mathrm{VC}(\mathcal{H}(i_1, \cdots, i_m)) \ge \mathrm{VC}(\mathcal{L}_m) \tag{18}$$

Note that $\mathcal{H} \subset \bigcup_{(i_1,i_2,\cdots,i_m)} \mathcal{H}(i_1, i_2, \cdots, i_m)$ because at most m parameters are allowed to change during surgery. The number of tuples $(i_1, i_2, \cdots, i_m)$ is $\binom{d}{m}$ because it is equivalent to choose m dimensions from d dimensions. Consider the growth function, according to Lemma 1 and Lemma 2,

$$\Pi_{\mathcal{H}}(n) \le \sum_{(i_1,i_2,\cdots,i_m)} (\Pi_{\mathcal{H}(i_1,i_2,\cdots,i_m)}(n)) \tag{19}$$

$$\le \binom{d}{m} \Pi_{\mathcal{L}_{m+1}}(n) \tag{20}$$

$$\le \binom{d}{m} \left(\frac{en}{m + 1}\right)^{m+1} \tag{21}$$

$$\le \left(\frac{d}{m}\right)^m \left(\frac{en}{m + 1}\right)^{m+1} \tag{22}$$

Define $n = \mathrm{VC}(\mathcal{H}), k = m + 1$,

$$2^n = \Pi_{\mathcal{H}}(n) \le \left(\frac{d}{m}\right)^m \left(\frac{en}{k}\right)^k \tag{23}$$

$$\le \left(\frac{d}{k}\right)^k \left(\frac{en}{k}\right)^k = \left(\frac{end}{k^2}\right)^k \tag{24}$$

[4] Please refer to more details about the lemma in the tutorial: https://www.cs.cmu.edu/~./awm/tutorials/vcdim08.pdf.

[5] Please refer to more details about the lemma in the wiki: https://en.wikipedia.org/wiki/Sauer%E2%80%93Shelah_lemma

Here $(\frac{d}{m})^m \leq (\frac{d}{k})^k$ holds when $k < \frac{d}{e}$ because $(\frac{d}{x})^x$ is increasing when $x < \frac{d}{e}$.

Define $r = \frac{n}{k}$ and take the logarithm,

$$n \leq k \log_2 \frac{edn}{k^2}, \quad r \leq \log_2 \left(\frac{edr}{k} \right) \qquad (25)$$

Define $f(t) = t - \log_2(\frac{ed}{k}) - \log_2 t$, we have $\frac{ed}{k} > e^2 > 4$ then $f(r) < 0$, since $\frac{k}{d} < \frac{1}{e}$, we have $f'(t) = 1 - \frac{1}{t \ln 2}$, when $t > \frac{1}{\ln 2}$, $f'(t) > 0$. Define $s = \log_2(\frac{ed}{k})$, we have $s > 2$, when $r > r_0 = 2s$,

$$f(r) > f(r_0) = 2s - s - \log_2(2s) > 0 \qquad (26)$$

Combined with $f(r) \leq 0$, we have $r \leq r_0$ and $n \leq 2(m+1)s$, that is

$$\text{VC}(\mathcal{H}) = n \leq 2(m+1)\log_2 \left(\frac{ed}{m+1} \right) \qquad (27)$$

To conclude, when $m < \frac{1}{e}d - 1 \approx 0.37d - 1$,

$$m \leq \text{VC}(\mathcal{H}) \leq 2(m+1)\log_2 \left(\frac{ed}{m+1} \right) \qquad (28)$$

□

A.2 Details of Datasets and Experiments

In this section, we introduce detailed dataset statistics and experimental settings. Experiments are conducted on a GeForce GTX TITAN X GPU.

A.2.1 Applications to Classification Problems

We conduct targeted backdoor learning experiments on fine-tuned BERT model on IMDB and SST-2.

IMDB and SST-2. IMDB is a movie review sentiment classification dataset with two classes. It includes 50000 training sentences and 50000 test sentences. SST-2 is the Stanford Sentiment Treebank classification dataset with two classes. It includes 63750 training sentences, 873 development sentences, and 1820 test sentences. In our paper, we adopt the development sentences as the test set. The sentences are preprocessed to lowercased and tokenized by the uncased BERT tokenizer. Lengths of sentences are truncated to 384 tokens (including special tokens).

Initial Model Implementation. The initial model is a fine-tuned uncased BERT base model. We adopt the AdamW optimizer. The training batch size is 8 and the learning rate is 2e-5. We fine-tuning the model for 10 epochs. The gradient norm is clipped to 1.0. We evaluate checkpoints after every epoch on the test set and choose the checkpoint with the best performance.

Experimental Settings. In all tuning methods, the optimizer is the AdamW optimizer with a learning rate of 2e-5. The training batch size is 8. The weight-decay is 5×10^{-4}. We train the model for 40000 iterations. The gradient norm is clipped to 1.0. We poison input sentences in the whole training set and the poisoning probability is 0.5. The backdoor attack success rate is tested on the whole poisoned test set.

Hyper-parameters Selection. In Sel-LCA surgery method, we choose $T = 2$ steps to estimate LCA. In dynamic surgery methods, we chose $K_{\text{start}} = 100, K_{\text{every}} = 30, \alpha = 0.97, \eta = 0.95$. For Lagrange surgery methods, we try λ in {0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.4, 0.6, 0.6, 0.8, 1.0}.

A.2.2 Applications to Generation Problems

We conduct neural network patching experiments on GRU-based sequence-to-sequence dialogue systems. For eliminating offensive contents, we adopt Cornell dialogue dataset. For injecting easter eggs, we adopt Daily dialogue dataset.

Cornell Dialogue and Daily Dialogue. Cornell Dialog consists of single-turn dialogues in movies. Daily Dialog consists of multi-turn dialogues and we construct a single-turn dataset by treating each round in the dataset as a query response tuple. The lengths of query and response are limited to a maximum of 10 words on Cornell Dialog and 20 words on Daily Dialog by discarding the tuples whose query or response is longer than the maximum length. Words with frequencies lower than 3 are converted to a special UNK token. Raw texts are preprocessed and lowercased. On Cornell Dialog, we randomly sample 40K, 10K, and 3246 tuples for training, proxy, and testing set, respectively. On Daily Dialog, we randomly sample 21.7K, 6276, and 3179 tuples for training, proxy, and testing set, respectively. Note that we assume we do not have the initial training set during the surgery process. Therefore, we use a proxy dataset instead. The training set is divided into two folds. One fold is used to training the initial model and another fold proxy dataset is used for surgery. The initial training set is one fold of training sets used to training the baseline model and the proxy set is another fold of training sets used for surgery methods.

Initial Model Implementation. The initial model is a GRU-based sequence-to-sequence model. The encoder and decoder are both 2-layer GRUs. The hidden size is 500 and the dropout rate is 0.1. The decoder adopts a global dot attention mechanism. We adopt the AdamW optimizer. The training batch size is 64 and the learning rate is 1e-4. We train the model for 60K iterations utilizing teacher forcing. The gradient norm is clipped to 50.0. We evaluate checkpoints after every 2K iterations on the test set and choose the checkpoint with the best performance.

Experimental Settings. In all tuning methods, we adopt the AdamW optimizer. The training batch size is 64 and the learning rate is 5e-5. We train the model for 20K iterations utilizing teacher forcing. The gradient norm is clipped to 50.0. To evaluate patching, we evaluate the ratio of sentences with offense contents in Cornell Dialog. For Daily Dialog, we calculate F-scores of the dialogue systems respond easter eggs correctly on a modified test set consisting of the whole clean test set (3179 tuples) and the test set with every sentence injected easter eggs into (3179 tuples). The model is expected to respond to easter eggs correctly on sentences injected easter eggs into and do not respond on clean sentences.

Human Evaluation Details. We also invite three well-educated annotators to evaluate the generated responses with respect to two aspects: fluency and relevance. Fluency indicates how likely the generated text is produced by a human. Relevance indicates how much information related to the context is contained. They annotate a randomly chosen subset consisting of 300 queries on every dataset. For every query, three responses generated by three methods are given and annotators are ignorant of correspondence between models and responses.

Hyper-parameters Selection. For Dyn-Δ surgery method, we chose $K_{\text{start}} = 50, K_{\text{every}} = 10, \alpha = 0.95, \eta = 0.95$.

A.2.3 Experiments Comparing Different Surgery Methods

We conduct targeted backdoor learning experiments on the ResNet-18 model on CIFAR-10.

CIFAR-10. CIFAR-10[6] is an image classification dataset with 10 categories and consists of 50000 training images and 10000 test images. The images are of 32-by-32 pixel size with 3 channels. We adopt the classification accuracy as our evaluation metric on CIFAR-10.

Initial Model Implementation. The initial model is ResNet-18. Following are settings when training the initial model. The optimizer is the SGD optimizer with a learning rate of 0.1 and a momentum of 0.9. The mini-batch size of 128. The weight-decay is 5×10^{-4}. We train the model for 200 epochs. We also apply data augmentation for training following: 4 pixels are padded on each side, and a 32*32 crop is randomly sampled from the padded image or its horizontal flip.

Experimental Settings. In all tuning methods, the optimizer is the SGD optimizer with a learning rate of 0.01 and a momentum of 0.9. The mini-batch size is 32. The weight-decay is 5×10^{-4}. We train the model for 200 epochs. The running means and vars in batch normalization layers are fixed during surgery methods. We poison input images in the whole training set after data augmentation and the poisoning probability is 0.5. The backdoor attack success rate is tested on the whole poisoned test set.

Hyper-parameters Selection. In Sel-LCA surgery method, we choose $T = 3$ steps to estimate LCA. In dynamic surgery methods, we chose $K_{\text{start}} = 100, K_{\text{every}} = 10, \alpha = 0.95, \eta = 0.9$. For Lagrange surgery methods, we try λ in {1e-4, 2e-4, 5e-4, 1e-3, 2e-3, 5e-3, 1e-2, 2e-2, 5e-2, 0.1, 0.2, 0.5, 1, 2, 5}.

A.3 Hyper-parameters Selection in Dynamic Surgery

In dynamic surgery methods. K_{start} are recommend to set as 50-100. $K_{\text{every}}, \alpha, \eta$ should be set according to the number of model parameters and training iterations. Suppose the model has N_p parameters and are trained K_{total} iterations, if the pruning process are expected to finish in ρK_{total} iterations, it is recommend that $\alpha^{K_{\text{every}}} \approx 0.5$ and $N_p \eta^{\rho * K_{\text{total}} / K_{\text{every}}} \approx 1$, we usually choose K_{every} in 10-50 and ρ in 0.25-0.5. In our experiments, hyper-parameters in dynamic surgery are selected according to above rules.

[6] CIFAR-10 can be found at https://www.cs.toronto.edu/~kriz/cifar.html