Robust Fine-tuning via Perturbation and Interpolation from In-batch Instances

Shoujie Tong\textsuperscript{1}, Qingxiu Dong\textsuperscript{1,}\textsuperscript{\ast}, Damai Dai\textsuperscript{1}, Yifan Song\textsuperscript{1}, Tianyu Liu\textsuperscript{1}, Baobao Chang\textsuperscript{1} and Zhifang Sui\textsuperscript{1}

\textsuperscript{1}Key Laboratory of Computational Linguistics, Peking University
\textsuperscript{2}Tencent Cloud Xiaowei
\{tong,dqx\}@stu.pku.edu.cn, \{daidai,yfsong,chbb,szf\}@pku.edu.cn, \{rogertyliu\}@tencent.com

Abstract

Fine-tuning pretrained language models (PLMs) on downstream tasks has become common practice in natural language processing. However, most of the PLMs are vulnerable, e.g., they are brittle under adversarial attacks or imbalanced data, which hinders the application of the PLMs on some downstream tasks, especially in safe-critical scenarios. In this paper, we propose a simple yet effective fine-tuning method called MATCH-TUNING to force the PLMs to be more robust. For each instance in a batch, we involve other instances in the same batch to interact with it. To be specific, regarding the instances with other labels as a perturbation, MATCH-TUNING makes the model more robust to noise at the beginning of training. While nearing the end, MATCH-TUNING focuses more on performing an interpolation among the instances with the same label for better generalization. Extensive experiments on various tasks in GLUE benchmark show that MATCH-TUNING consistently outperforms the vanilla fine-tuning by 1.64 scores. Moreover, MATCH-TUNING exhibits remarkable robustness to adversarial attacks and data imbalance.

1 Introduction

Pretrained language models (PLMs) have contributed to striking success in natural language processing (NLP). Simultaneously, fine-tuning has been a common practice to employ PLMs for downstream natural language understanding tasks. However, recent work shows that vanilla fine-tuning methods may lead to vulnerable models [Aghajanyan et al., 2021]. This long-standing problem hinders the model performance and makes fine-tuned PLMs vulnerable to adversarial attacks and spurious bias [Branco et al., 2021; Clark et al., 2019]. As a result, it limits the application of the PLMs on some downstream tasks, especially in some real-world scenarios where robustness is especially required.

To alleviate this problem, various fine-tuning approaches have been proposed. For instance, SMART [Jiang et al., 2020] and R3F [Aghajanyan et al., 2021] introduce regularizations to the noise applied to the original pretrained representations. ChildTuning [Xu et al., 2021] updates the child network during fine-tuning via strategically masking out the gradients of the non-child network. However, most of them focus on improving the generalizing robustness or the adaptive robustness, while the robustness to adversarial attacks and spurious correlations remains challenging.

Inspired by contrastive learning with in-batch instances [Gao et al., 2021; Fan et al., 2021], we propose MATCH-TUNING that utilizes in-batch instances dynamically for robust fine-tuning. In MATCH-TUNING, we convert each instance representation in a batch to a matching matrix. Then, we fuse the PLM representations according to the matching matrix to form new representations for this batch. Finally, we use the new representations for prediction.

MATCH-TUNING works by adaptively determining how to utilize the in-batch instances during the whole training procedure. As shown in Fig. 1, at the beginning of training, regarding the instances with other labels as a perturbation, MATCH-TUNING urges the PLM to converge to a more flat local minimum for better robustness. While nearing the end, MATCH-TUNING focuses more on performing an interpolation among the instances with the same label for better generalization. In this manner, MATCH-TUNING reduces the vulnerability of models and improves their general performance.

We conduct a comprehensive evaluation of MATCH-TUNING on the GLUE benchmark. The results show that our method outperforms the vanilla fine-tuning by 1.64 scores on average. In addition, our method outperforms vanilla fine-tuning by 4.11 average scores on advGLUE, and yields a great improvement for label noise or data imbalance, which shows our overwhelming robustness over vanilla fine-tuning.

Our main contributions are summarized as follows:

\begin{itemize}
  \item We propose an adaptive fine-tuning method called MATCH-TUNING to train robust models, where instances in the same batch will interact with each other.\footnote{Equal contribution.}
  \item MATCH-TUNING reduces the vulnerability of models
\end{itemize}
and outperforms the vanilla fine-tuning by 1.64 scores on the GLUE benchmark.

• Our method manifests extraordinary robustness to various scenarios, including adversarial attacks, spurious biases, and data imbalance.

2 Related Work

The vanilla fine-tuning simply adapts PLMs to the task-specific inputs and outputs, and fine-tunes all the parameters in an end-to-end manner [Devlin et al., 2019; Liu et al., 2019a]. The token representations or the representation of a special token (e.g., [CLS]) is directly fed into an output layer for tagging or classification, respectively. This manner has been shown to produce biased models that are vulnerable to adversarial attacks and noisy data [Aghajanyan et al., 2021; Clark et al., 2019].

In the past years, numerical variants like ChildTuning and R3F are proposed to conduct more trustworthy and effective fine-tuning [Lee et al., 2020; Xu et al., 2021; Aghajanyan et al., 2021]. FreeLB [Zhu et al., 2019] adds perturbations to continuous word embedding by using a gradient method and minimizes the resultant adversarial risk. Moreover, Jiang et al. [Jiang et al., 2020] introduce a regularization to encourage the model output not to change much when injecting a small perturbation to the input. Aghajanyan et al. [Aghajanyan et al., 2021] simply regularize the model against the parametric noise. The literature provides strong insights that proper perturbation on the PLM outputs has great potential in enforcing the model smoothness and robustness.

Recently, in-batch learning successes in many fields. Yao et al. [Liu et al., 2019b; Yao et al., 2021] suggest that the unsupervised instances are helpful to learning the classifier in computer vision tasks. For NLP, contrastive learning with in-batch instances also improves the task-specific representation [Gao et al., 2021] and adversarial robustness [Fan et al., 2021]. Inspired by this, we propose MATCH-TUNING to utilize in-batch instances dynamically for robust fine-tuning. Different from previous work, MATCH-TUNING no longer needs label information or the specification of negative and positive instances in advance. It performs automatic instance interaction, which applies to most existing pretrained models.

3 Method

MATCH-TUNING derives a composite representation for each instance in a batch by fusing the representations of other instances in the same batch, which are deemed as adaptive noise. From experiments, we find that the adaptive noise functions as an in-batch perturbation in the initial stage of training, and then gradually transits to an in-batch interpolation among “positive” instances that share the same label. In addition, we show that MATCH-TUNING helps the model to escape the sharp local minima through qualitative analysis.

3.1 Overview of MATCH-TUNING

In the batched gradient descent, the data points in a batch are formulated as \(\{(x_i, y_i)\}_{i=1}^n\), where \(n\) denotes the batch size. Throughout our paper, \(x_i\) represents a textual input, e.g., a single sentence or a sentence pair, while \(y_i\) denotes a discrete label or a continuous number for classification and regression tasks, respectively. We use \(h\) and \(\theta\) to represent a PLM that extracts contextualized features from \(x_i\) and its parameters. Similarly, the task-specific classifier and its parameters are denoted by \(f\) and \(\psi\). Letting \(L\) denote the task-specific loss function, we compute the mini-batch gradient \(g\) in the vanilla fine-tuning as follows:

\[
g = \frac{1}{n} \nabla_\theta \frac{1}{n} \sum_{i=1}^n L \left(f(h(x^{(i)}; \theta); \psi), y^{(i)}\right). \tag{1}
\]

To apply adaptive weights to the instances in a batch, we introduce the matching matrix \(M\), where each element indicates the pair-level similarity between in-batch instance representations given by a PLM. The matrix \(M\) is given by

\[
M_{i,j} = \frac{\exp \left(h(x^{(i)}; \theta)h(x^{(j)}; \theta)\right)}{\sum_{k=1}^n \exp \left(h(x^{(i)}; \theta)h(x^{(k)}; \theta)\right)}. \tag{2}
\]
Note that the overhead to compute $M_{i,j}$ is small since we can directly reuse $h(x; \theta)$, the outputs of the PLM. Then, to produce a robust representation for an instance, we derive a composite representation $z^{(i)}$ from $h(x^{(i)}; \theta)$:

$$z^{(i)} = \sum_{j=1}^{n} M_{i,j} h(x^{(j)}; \theta).$$  

(3)

Then, $z^{(i)}$ serves as a drop-in replacement for $h(x^{(i)}; \theta)$ in the vanilla fine-tuning and the mini-batch gradient $g'$ in MATCH-TUNING is computed as follows:

$$g' = \frac{1}{n} \nabla_{\theta, \psi} \sum_{i=1}^{n} L \left( f(z^{(i)}; \psi), y^{(i)} \right).$$  

(4)

### 3.2 Qualitative Understanding of MATCH-TUNING

We provide a qualitative viewpoint to understand how MATCH-TUNING works stemming from the notions of perturbation and interpolation.

We first introduce the shape of local minima. The loss surface of deep neural networks tends to have various local minima as illustrated in Fig. 2. Sharp local minima are where the loss in a small neighborhood increase rapidly while flat local minima are where the loss varies slowly in a relatively large neighborhood. Sufficient literature has proved that flat local minima usually lead to better generalization [Hochreiter and Schmidhuber, 1995; Dinh et al., 2017]. In addition, under a Bayesian perspective, the noise in gradient could drive model away from sharp minima [Smith et al., 2018].

MATCH-TUNING follows this gradient noise addition routine, but in a different way. To better understand the mechanism of MATCH-TUNING, we visualize the values of the “positive” instances (with the same label) and “negative” instances (with other labels) in the matching matrix. Fig. 3 depicts the change of matching matrix in MATCH-TUNING. We sum up the values of all instances with the same label and show the cumulative values for “positive” instances on Fig. 3. The “negative” instances share the same setting.

**In-batch Perturbation** In the initial stage, MATCH-TUNING works by performing in-batch perturbation on the original outputs of PLM. As shown in Fig. 3, instances from the same class and that from other classes are similar in the matching matrix. Therefore, any other instance in the same batch is close to a tiny perturbation on the output of PLM. If the PLM provides vulnerable representations, the perturbed representations will break down easily. Therefore, the early stage in MATCH-TUNING encourages the PLM to generate a more robust representation for each instance and converge to a more flat local minimum.

**In-batch Interpolation** During the whole training process, the model will gradually learn to distinguish the representations of the positive (same label) and negative (other labels) instances in a batch. Consequently, as training proceeds, the portion of negative instances is getting smaller in the matching matrix and can hardly influence the composite representation in the late stage of training. In this moment, MATCH-TUNING tends to interpolate the representations of the positive instances. We also observe more representative positive instances will contribute more to the final composite representation. As illustrated in Fig. 1, the late stage in MATCH-TUNING encourages composite representations to be grouped into clusters according to their real labels.

### 4 Experiments

We conducted extensive experiments on various downstream tasks to evaluate the general performance and robustness of MATCH-TUNING. For simplicity, in the rest of the paper, we denote other instances in the same batch with the same label as the current instance by positive instances, and other instances with different labels by negative samples.

#### 4.1 Datasets

Following previous work [Xu et al., 2021; Lee et al., 2020], we conduct experiments on four main datasets in GLUE [Wang et al., 2019] to evaluate the general performance. Among them, classification task like CoLA is for linguistic acceptability, RTE is for natural language inference, and MRPC is for paraphrase identification. STS-B is a regression task for semantic textual similarity. By systematically conducting 14 kinds of adversarial attacks on representative instances.
GLUE tasks, Wang et al. [Wang et al., 2021] proposed AdvGLUE, a multi-task benchmark to evaluate and analyze the robustness of language models and robust training methods.

4.2 Experimental Setup

We report the averaged results over 10 random seeds. We conduct our experiments based on the HuggingFace transformers library\(^1\) and follow the default hyper-parameters and settings unless noted otherwise.

4.3 General Performance

We compare MATCH-TUNING with the vanilla fine-tuning and related work on four tasks of the well-recognized benchmark, GLUE. And we focus on evaluating the performance of BERT-Large based models on the GLUE development set.

| Method                  | CoLA 74.12 (76.17) | RTE 73.63 (76.17) | MRPC 91.70 (92.39) | STS-B 90.45 (90.89) | Avg 80.17 | \(\Delta\) |
|-------------------------|---------------------|-------------------|---------------------|---------------------|-----------|-----------|
| Vanilla Fine-tuning     | 63.16 (64.55)       | 70.61 (74.37)     | 90.70 (91.42)       | 89.64 (90.99)       | 78.53     | 0.00      |
| Weight Decay [Daumé III, 2007] | 63.26 (64.76)       | 72.10 (74.77)     | 90.88 (91.62)       | 89.66 (90.22)       | 78.98     | +0.45     |
| Top-K Tuning [Houlsby et al., 2019] | 63.02 (63.88)       | 70.92 (74.37)     | 91.04 (92.23)       | 89.64 (90.83)       | 78.66     | +0.13     |
| Mixout [Lee et al., 2020] | 63.78 (65.55)       | 72.32 (75.52)     | 91.19 (92.01)       | 89.89 (90.33)       | 79.30     | +0.77     |
| RecAdam [Chen et al., 2020] | 63.99 (65.53)       | 71.82 (73.30)     | 90.84 (91.89)       | 89.67 (90.42)       | 79.08     | +0.55     |
| R3F [Aghajanyan et al., 2021] | 64.03 (66.24)       | 72.42 (74.37)     | 91.09 (91.32)       | 89.64 (90.99)\(^*\) | 79.30     | +0.77     |
| ChildTuning [Xu et al., 2021] | 63.70 (66.12)       | 72.02 (74.17)     | 91.23 (92.01)       | 90.16 (90.68)       | 79.28     | +0.75     |
| ChildTuning\(D\) [Xu et al., 2021] | 64.84 (66.17)       | 73.23 (76.17)     | 91.42 (92.20)       | 90.18 (90.88)       | 79.92     | +1.39     |
| MATCH-TUNING            | 64.39 (67.25)       | 74.12 (76.17)     | 91.70 (92.39)       | 90.45 (90.89)       | 80.17     | +1.64     |
| MATCH-TUNING + R3F      | 65.21 (67.25)       | 73.63 (76.17)     | 92.34 (93.22)       | 90.45 (90.89)\(^*\) | 80.41     | +1.88     |

Table 1: Comparison between MATCH-TUNING with other fine-tuning methods. We report the mean (max) results of 10 random seeds. Note that since R3F is not applicable to regression task, the results on STS-B (marked with \(^*\)) remain the same as vanilla and MATCH-TUNING, respectively. MATCH-TUNING achieves the best performance compared with other methods. Integrating MATCH-TUNING with other fine-tuning methods like R3F can yield further improvements.

Recent work reveals that vanilla fine-tuning is deceptive and vulnerable in many aspects. For instance, fool the models to output arbitrarily wrong answers by perturbing input sentences in a human-imperceptible way. Real-world systems built upon these vulnerable models can be misled in ways that would have profound security concerns. To examine the robustness of MATCH-TUNING, we design robustness evaluation tasks for three common scenarios respectively.

4.4 Robustness of MATCH-TUNING

Recent work reveals that vanilla fine-tuning is deceptive and vulnerable in many aspects. For instance, fool the models to output arbitrarily wrong answers by perturbing input sentences in a human-imperceptible way. Real-world systems built upon these vulnerable models can be misled in ways that would have profound security concerns. To examine the robustness of MATCH-TUNING, we design robustness evaluation tasks for three common scenarios respectively.

Robustness to Adversarial Attacks

As recent studies revealed, the robustness of fine-tuned PLMs can be challenged by carefully crafted textual adversarial examples. We systematically conduct various adversarial attack evaluations on the advGLUE benchmark.

Tab. 2 illustrates that fine-tuned models maintain vulnerabilities to adversarial attacks while our MATCH-TUNING approach alleviates this chronic problem by 4.11 accuracy promotion on average. Compared with vanilla fine-tuning, existing methods like R3F and ChildTuning encounter 8 ~ 13 accuracy collapse on advSST-2, while MATCH-TUNING outperforms vanilla fine-tuning by 4.11 scores. On advMNLI, advRTE, and advQQP, MATCH-TUNING also holds a large improvement, as much as 10.79 higher accuracy than vanilla fine-tuning. In short, compared with prior fine-tuning methods, we find that MATCH-TUNING is more robust in adapting PLMs to various tasks.

Robustness to Label Noise

Over the past decade, there is inevitably some noise in large-scale datasets. To explore the model robustness to noisy data, we conduct simple simulation experiments on RTE, MRPC, and CoLA. Specifically, we generate noisy training data by randomly changing a certain proportion of labels to incorrect

---

\(^1\)https://github.com/huggingface/transformers
We test the robustness of different fine-tuning methods trained on the noisy data.

As shown in Tab. 3, MATCH-TUNING outperforms other fine-tuning methods on noisy training data. To be exact, MATCH-TUNING surpasses vanilla fine-tuning by 1.76 average scores under 5% noise ratio, 2.56 under 10% noise ratio, and 1.41 under 15% noise ratio. Furthermore, we compare the degradation of model performance towards different noise ratios. Compared with Tab. 1, we calculated the degradation and display it in brackets (the last column of the Tab. 3 ). It shows that MATCH-TUNING has the smallest performance drop compared to other fine-tuning methods. All the above results show that MATCH-TUNING is more robust to label noise than existing methods.

Robustness to Data Imbalance

Minority Class Minority class refers to the class which owns insufficient instances in the training set. These kinds of classes are more challenging during fine-tuning than a normal class. To explore the performance of different tuning approaches on the minority class, we conduct experiments on synthetic RTE, MRPC, and CoLA datasets.

As Tab. 5 illustrated, under different data reduction ratios, MATCH-TUNING outperforms other fine-tuning methods by a large margin. MATCH-TUNING yields an improvement of up to 6.12 average score on 30% reduction ratio and 4.89 average scores on 40% reduction ratio. Besides, it can be seen that the smaller the reduction ratio, the better MATCH-TUNING performs compared to other fine-tuning methods. In summary, we can conclude that MATCH-TUNING is more robust towards the minority class.

Atypical Groups Vanilla fine-tuned models can be highly accurate on average on an i.i.d. test set yet consistently fail on atypical groups of the data [Hovy and Søgaard, 2015; Sagawa et al., 2019] (e.g., by learning spurious correlations that hold on average but not in such groups). In contrast, MATCH-TUNING no longer aims at minimizing the original batch average loss and paying more attention to the comparison of instances. As demonstrated in Tab. 6, simply applying MATCH-TUNING improves the worst-group performance by 1.1 with traditional empirical risk minimization (ERM) and 1.5 with GroupDRO [Sagawa et al., 2019]. What’s more, the results show that MATCH-TUNING is orthogonal to prior techniques for data imbalance, integrating MATCH-TUNING with them brings further improvement.

5 Analysis and Discussion

5.1 Exploration into Effects of In-batch Instances

As analyzed in Section 3.2, negative instances and positive instances in a batch function differently in the process of MATCH-TUNING. To further explore the role of negative instances and positive instances in MATCH-TUNING, we define a mask matrix $A$ by:

$$A_{i,j} = \begin{cases} 1, & y^{(i)} = y^{(j)} \\ 0, & y^{(i)} \neq y^{(j)} \end{cases}$$

Then we update $M \leftarrow A \odot M$ so that merely positive instances are involved for MATCH-TUNING, while $M \leftarrow (I_n - A) \odot M$ so that only negative instances are observed.

As is shown in Tab. 7, both negative instances and positive instances play an important role in MATCH-TUNING. When negative instances are masked for the matching matrix and only positive instances are responsible for MATCH-TUNING, the resulting score outperforms on RTE and MRPC, but drops on CoLA slightly. In the contrast, if we only preserve the influence of negative instances on the current instance, the performance surpasses the vanilla fine-tuning baseline steadily.

---

Table 2: Robustness evaluation on AdvGLUE validation set. We report the mean results of 3 random seeds. MATCH-TUNING achieves considerable improvement on most datasets, especially on the SST-2 and RTE datasets.

| Method         | advSST-2 | advMNLI | advRTE | advQNLI | advQQP | Avg | Δ     |
|----------------|----------|---------|--------|---------|--------|-----|-------|
| Vanilla Fine-tuning | 47.57    | 34.99/30.00 | 41.73  | 46.40   | 38.45/27.59 | 40.24 | 0.00  |
| R3F [Aghajanyan et al., 2021] | 38.51    | 35.81/30.26 | 50.12  | 47.52   | 40.59/35.23 | 41.42 | +1.18 |
| ChildTuning $F$ [Xu et al., 2021] | 34.46    | 33.88/26.53 | 41.98  | 47.53   | 40.38/35.82 | 38.46 | -1.78 |
| ChildTuning $D$ [Xu et al., 2021] | 39.19    | 34.06/27.84 | 46.17  | 49.55   | 40.66/39.80 | 41.22 | +0.98 |
| MATCH-TUNING | **51.35** | **35.54/31.07** | **52.52** | **47.52** | **41.45/32.62** | **44.35** | +4.11 |

Table 3: Comparison of different tuning approaches on robustness towards label noise. The noise ratio refers to the proportion of training instances whose labels are transferred to incorrect labels. MATCH-TUNING can maintain more robust representations compared with other fine-tuning methods.

| Method         | Noise Ratio  | CoLA       | MRPC       | RTE        | Avg       | Δ     |
|----------------|--------------|------------|------------|------------|-----------|-------|
| Vanilla Fine-tuning | 5%          | 61.14      | 90.38      | 69.68      | 73.73 (↓ 4.80) |       |
| R3F             | 5%          | 62.42      | 90.82      | 67.99      | 73.74 (↓ 5.56) |       |
| ChildTuning $F$ | 5%          | 61.13      | 90.46      | 71.59      | 74.39 (↓ 4.89) |       |
| ChildTuning $D$ | 5%          | 61.46      | 90.42      | 71.72      | 74.53 (↓ 5.39) |       |
| MATCH-TUNING    | 5%          | 62.33      | 91.19      | 72.96      | 75.49 (↓ 4.68) |       |
| Noise Ratio 10% |             |            |            |            |           |       |
| Vanilla Fine-tuning | 10%         | 59.21      | 88.90      | 68.34      | 72.15 (↓ 6.38) |       |
| R3F             | 10%         | 61.76      | 90.36      | 66.75      | 72.96 (↓ 6.34) |       |
| ChildTuning $F$ | 10%         | 60.97      | 89.83      | 70.02      | 73.61 (↓ 5.67) |       |
| ChildTuning $D$ | 10%         | 61.35      | 89.92      | 69.06      | 73.44 (↓ 6.84) |       |
| MATCH-TUNING    | 10%         | 61.41      | 90.55      | 71.56      | 74.51 (↓ 5.66) |       |
| Noise Ratio 15% |             |            |            |            |           |       |
| Vanilla Fine-tuning | 15%         | 59.01      | 87.84      | 68.12      | 71.66 (↓ 6.87) |       |
| R3F             | 15%         | 60.16      | 88.51      | 65.14      | 71.27 (↓ 8.03) |       |
| ChildTuning $F$ | 15%         | 59.66      | 88.08      | 69.10      | 72.26 (↓ 7.00) |       |
| ChildTuning $D$ | 15%         | 59.88      | 89.01      | 69.78      | 72.89 (↓ 7.03) |       |
| MATCH-TUNING    | 15%         | 59.65      | 89.51      | 70.04      | 73.07 (↓ 7.10) |       |
Besides, since R3F is not applicable to the regression task, the result marked with * introduce heavy extra computational cost, M\_A40 GPU. As illustrated in Tab.4, while other methods all are based on BERT\_LARGE for different fine-tuning methods. All the methods actions. To demonstrate the computational efficiency of MATCH-\textsc{Tuning} contribute to the final improvements. M\_\textsc{Tuning} takes almost no overhead than vanilla fine-tuning. Besides, this result indicates that both perturbation and interpolation contribute to the final improvements. MATCH-\textsc{Tuning} simply unify negative instances and positive instances by the matching matrix, and such unification brings further improvement (refer to the last row of Tab. 7).

### 5.2 Computational Efficiency

MATCH-\textsc{Tuning} improves the general performance and robustness of PLMs by introducing simple in-batch interactions. To demonstrate the computational efficiency of MATCH-\textsc{Tuning}, we report the training time of a single epoch for different fine-tuning methods. All the methods are based on BERT\_LARGE and tested on a single NVIDIA A40 GPU. As illustrated in Tab.4, while other methods all introduce heavy extra computational cost, MATCH-\textsc{Tuning} takes almost no overhead than vanilla fine-tuning. Besides, this result indicates that both perturbation and interpolation contribute to the final improvements. MATCH-\textsc{Tuning} simply unify negative instances and positive instances by the matching matrix, and such unification brings further improvement (refer to the last row of Tab. 7).

### 6 Conclusions

To improve the general performance and robustness for fine-tuning PLMs, we propose robust MATCH-\textsc{Tuning} via in-batch instance perturbation. Extensive experiments on downstream tasks demonstrate the general performance of MATCH-\textsc{Tuning}. In addition, MATCH-\textsc{Tuning} is shown to be a powerful tuning approach towards broad categories of robustness evaluation. We further analyze the functioning process of MATCH-\textsc{Tuning} and provide probation on its components.

### Acknowledgements

This paper is supported by the National Key Research and Development Program of China 2020AAA0106700 and NSFC project U19A2065.
References

[Aghajanyan et al., 2021] Armen Aghajanyan, Akshat Shrivastava, Anicht Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. Better fine-tuning by reducing representational collapse. In International Conference on Learning Representations (ICLR), 2021.

[Branco et al., 2021] Ruben Branco, António Branco, João Rodrigues, and João Silva. Shortcutted commonsense: Data spuriousness in deep learning of commonsense reasoning. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 1504–1521, 2021.

[Chen et al., 2020] Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, EMNLP, 2020.

[Clark et al., 2019] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Don’t Take the Easy Way Out: Ensemble Based Methods for Avoiding Known Dataset Biases. arXiv, 2019.

[Daumé III, 2007] Hal Daumé III. Frustratingly easy domain adaptation. In Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL), June 2007.

[Devlin et al., 2017] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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), June 2019.

[Dinh et al., 2017] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In International Conference on Machine Learning, pages 1019–1028. PMLR, 2017.

[Fan et al., 2021] Lijie Fan, Sijia Liu, Pin-Yu Chen, Gaoyuan Zhang, and Chuang Gan. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? Advances in Neural Information Processing Systems, 34, 2021.

[Gao et al., 2021] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. arXiv preprint arXiv:2104.08821, 2021.

[Hochreiter and Schmidhuber, 1995] Sepp Hochreiter and Jürgen Schmidhuber. Simplifying neural nets by discovering flat minima. In Advances in neural information processing systems, pages 529–536, 1995.

[Houlsby et al., 2019] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In ICML 2019, 2019.

[Hovy and Søgaard, 2015] Dirk Hovy and Anders Søgaard. Tagging performance correlates with author age. In ACL, pages 483–488, 2015.

[Jiang et al., 2020] Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tao Zhao. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), July 2020.

[Lee et al., 2020] Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. In ICLR, 2020.

[Liu et al., 2019a] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692, 2019.

[Liu et al., 2019b] Yufan Liu, Jiajiong Cao, Bing Li, Chunfeng Yuan, Weiming Hu, Yangxi Li, and Yunqiang Duan. Knowledge distillation via instance relationship graph. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7096–7104, 2019.

[Sagawa et al., 2019] Shiori Sagawa, Pang Wei Koh, Tatsumori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. arXiv preprint arXiv:1911.08731, 2019.

[Smith et al., 2018] Samuel L Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V Le. Don’t decay the learning rate, increase the batch size. In International Conference on Learning Representations, 2018.

[Wang et al., 2019] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In ICLR, 2019.

[Wang et al., 2021] Boxin Wang, Chejian Xu, Shuohang Wang, Zhe Gan, Yu Cheng, Jianfeng Gao, Ahmed Hassan Awadallah, and Bo Li. Adversarial glue: A multi-task benchmark for robustness evaluation of language models. arXiv preprint arXiv:2111.02840, 2021.

[Xu et al., 2021] Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanki Tan, Baobao Chang, Songfang Huang, and Fei Huang. Raise a child in large language model: Towards effective and generalizable fine-tuning. In EMNLP, pages 9514–9528, 2021.

[Yao et al., 2021] Yu Yao, Tongliang Liu, Mingming Gong, Bo Han, Gang Niu, and Kun Zhang. Instance-dependent label-noise learning under a structural causal model. Advances in Neural Information Processing Systems, 34, 2021.

[Zhu et al., 2019] Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. Freelb: Enhanced adversarial training for natural language understanding. arXiv preprint arXiv:1909.11764, 2019.