# Supplementary Materials
## *Transferable Query Selection for Active Domain Adaptation*

Bo Fu *, Zhangjie Cao *, Jianmin Wang, Mingsheng Long (✉)
School of Software, BNRist, Tsinghua University, China
{microhhh9,caozhangjie14}@gmail.com, {jimwang, mingsheng}@tsinghua.edu.cn

In the supplementary materials, we will provide the theoretical insight of active domain adaptation, more details on the implementation, and more experimental results.

## 1. Theoretical Insight of Active Domain Adaptation

The seminal work on domain adaptation theory [9] provides a bound on the target expected error $\epsilon_t$ as follows:

**Theorem 1** *[9] Let $\langle \mathcal{D}_s, l_s \rangle$ and $\langle \mathcal{D}_t, l_t \rangle$ be the source and target domains, where $l_s$ and $l_t$ are the optimal labeling functions. For any binary function class $\mathcal{H} \in [0,1]^{\mathcal{X}}$, and $\forall h \in \mathcal{H}$, the following generalization bound holds:*

$$\begin{aligned}\epsilon_t(h) \leq \epsilon_s(h) + d_{\tilde{\mathcal{H}}}(\mathcal{D}_s, \mathcal{D}_t) \\ + \min\{\mathbb{E}_{\mathcal{D}_s}[|l_s - l_t|], \mathbb{E}_{\mathcal{D}_t}[|l_s - l_t|]\},\end{aligned} \tag{1}$$

*where $\epsilon_t(h)$ and $\epsilon_s(h)$ are target and source expected error of hypothesis $h$ respectively. $\tilde{\mathcal{H}} = \{\text{sgn}(|h(\mathbf{x}) - h'(\mathbf{x}) - t|) | h, h' \in \mathcal{H}, 0 < t < 1\}$ is loss-derived hypothesis space and $d_{\tilde{\mathcal{H}}}(\mathcal{D}_s, \mathcal{D}_t)$ is the marginal domain discrepancy.*

In this paper, hypothesis $h$ is defined as $h = C(F(x))$. Previous unsupervised domain adaptation methods usually suppose that jointly minimizing the first two terms $\epsilon_s(h)$ and $d_{\tilde{\mathcal{H}}}(\mathcal{D}_s, \mathcal{D}_t)$ is sufficient for unsupervised domain adaptation, because they assume there exists an optimal common labeling function on the source and target domains to make the third term small enough or otherwise it is not proper to perform domain adaptation. They typically adopt a feature extractor $F$ to minimize the domain discrepancy. Thus, for their case, the terms $\mathcal{D}_s$ and $\mathcal{D}_t$ in the bound should refer to the distribution of features generated by $F$.

However, a new theory [9] shows that even if we can achieve low or zero $\min\{\mathbb{E}_{\mathcal{D}_s}[|l_s - l_t|], \mathbb{E}_{\mathcal{D}_t}[|l_s - l_t|]\}$ with the original input space, after a feature transformation $F$, the difference between optimal labeling functions can be enlarged because the new labeling functions and expectations are defined on source and target features but not original inputs. Let $h$ be the prediction function of the current training model. To ensure that the feature extractor $F$ does not increase the third term in Equation (1), we can derive an upper bound for it based on the sub-additivity:

$$\begin{aligned}&\min\{\mathbb{E}_{\mathcal{D}_s}[|l_s - l_t|], \mathbb{E}_{\mathcal{D}_t}[|l_s - l_t|]\} \\ &\leq \min\{\mathbb{E}_{\mathcal{D}_s}[|l_s - h| + |l_t - h|], \\ &\qquad \mathbb{E}_{\mathcal{D}_t}[|l_s - h| + |l_t - h|]\} \\ &\leq \mathbb{E}_{\mathcal{D}_s}|l_s - h| + \mathbb{E}_{\mathcal{D}_s}|l_t - h| \\ &+ \mathbb{E}_{\mathcal{D}_t}|l_s - h| + \mathbb{E}_{\mathcal{D}_t}|l_t - h|\end{aligned} \tag{2}$$

where $|l_s - h|$ and $|l_t - h|$ measures the discrepancy between the current prediction function and the source/target optimal labeling functions. And note that $l_s$ and $l_t$ can only be well-defined on $\mathcal{D}_s$ and $\mathcal{D}_t$ respectively.

By split, $\mathbb{E}_{\mathcal{D}_t}|l_s - h| = \mathbb{E}_{\mathcal{D}_t \cap \mathcal{D}_s}|l_s - h| + \mathbb{E}_{\mathcal{D}_t \setminus \mathcal{D}_s}|l_s - h|$. We delve into the term $\mathbb{E}_{\mathcal{D}_t \setminus \mathcal{D}_s}|l_s - h|$. Note that $l_s$ is the source *labeling function*, which can only give the ground-truth labels for samples in $\mathcal{D}_s$ but is undefined for samples out of $\mathcal{D}_s$. In other words, we can replace $l_s$ by proper function for samples out of $\mathcal{D}_s$. Therefore, we define $l_s = h$ for samples in $\mathcal{D}_t \setminus \mathcal{D}_s$. Then we have $\mathbb{E}_{\mathcal{D}_t \setminus \mathcal{D}_s}|l_s - h| = 0$. So $\mathbb{E}_{\mathcal{D}_t}|l_s - h| = \mathbb{E}_{\mathcal{D}_t \cap \mathcal{D}_s}|l_s - h|$. Similarly, we define $l_t = h$ for samples in $\mathcal{D}_s \setminus \mathcal{D}_t$, and $\mathbb{E}_{\mathcal{D}_s}|l_t - h| = \mathbb{E}_{\mathcal{D}_s \cap \mathcal{D}_t}|l_t - h|$.

Since we train $h$ to minimize the classification error on source data, $h$ is expected to be close to $l_s$ on $\mathcal{D}_s$. So both $\mathbb{E}_{\mathcal{D}_s}|l_s - h|$ and $\mathbb{E}_{\mathcal{D}_t \cap \mathcal{D}_s}|l_s - h|$ should be small. To further ensure that $\mathbb{E}_{\mathcal{D}_s \cap \mathcal{D}_t}|l_t - h|$ and $\mathbb{E}_{\mathcal{D}_t}|l_t - h|$ are small, a direct way is to annotate target samples that are (1) of high uncertainties or of wrong predictions by the current classifier and (2) to cover the whole target domain and incorporate them into supervised training. The two conditions are sufficiently fulfilled by $\text{TQS}_u$, $\text{TQS}_c$ and $\text{TQS}_d$. Hence, our transferable query selection metric can potentially bound the third term of the target error bound. Our approach is orthogonal and complementary to previous UDA methods based on domain discrepancy minimization. Integrating our approach with them can ensure a lower target error for domain adaptation.

*Equal contribution

Table 1. Classification accuracy on **Office-31** with 10% target samples as the the labeling budget for active methods.

| Method | A→D | A→W | D→A | D→W | W→A | W→D | Avg |
|---|---|---|---|---|---|---|---|
| ResNet | 81.5±0.1 | 75.0±0.1 | 63.1±0.2 | 95.2±0.1 | 65.7±0.1 | 99.4±0.1 | 80.0±0.1 |
| RAN | 90.9±0.5 | 90.4±0.5 | 80.4±0.6 | 98.3±0.3 | 80.8±0.5 | 99.6±0.1 | 90.1±0.5 |
| UCN | 94.0±0.3 | 93.0±0.3 | 84.0±0.4 | 100.0±0.0 | 84.3±0.4 | 100.0±0.0 | 92.6±0.3 |
| QBC | 93.3±0.2 | 93.1±0.2 | 83.0±0.3 | 99.5±0.2 | 84.2±0.2 | 99.6±0.1 | 92.1±0.2 |
| Cluster | 88.1±0.2 | 86.0±0.1 | 76.2±0.2 | 98.3±0.1 | 77.4±0.2 | 99.6±0.1 | 87.6±0.1 |
| AADA | 93.5±0.3 | 93.1±0.3 | 83.2±0.4 | 99.7±0.1 | 84.2±0.3 | 100.0±0.0 | 92.3±0.3 |
| ADMA | 94.0±0.2 | 93.4±0.3 | 84.4±0.3 | 100.0±0.0 | 84.6±0.3 | 100.0±0.0 | 92.7±0.3 |
| TQS | **96.4±0.3** | **96.4±0.3** | **86.4±0.4** | **100.0±0.0** | **87.1±0.3** | **100.0±0.0** | **94.4±0.3** |

Table 2. Classification accuracy on **Office-31** with 20% target samples as the the labeling budget for active methods.

| Method | A→D | A→W | D→A | D→W | W→A | W→D | Avg |
|---|---|---|---|---|---|---|---|
| ResNet | 81.5±0.1 | 75.0±0.1 | 63.1±0.2 | 95.2±0.1 | 65.7±0.1 | 99.4±0.1 | 80.0±0.1 |
| RAN | 94.6±0.3 | 94.4±0.4 | 86.0±0.5 | 99.5±0.3 | 83.6±0.4 | 99.4±0.1 | 92.9±0.4 |
| UCN | 98.0±0.2 | 98.1±0.2 | 90.8±0.2 | 100.0±0.0 | 90.2±0.2 | 100.0±0.0 | 96.2±0.2 |
| QBC | 97.7±0.1 | 97.4±0.1 | 90.0±0.1 | 100.0±0.0 | 89.0±0.2 | 100.0±0.0 | 95.7±0.1 |
| Cluster | 96.2±0.1 | 96.1±0.1 | 89.4±0.1 | 99.5±0.1 | 87.3±0.2 | 99.6±0.1 | 94.7±0.1 |
| AADA | 98.0±0.1 | 97.8±0.1 | 91.0±0.2 | 99.5±0.1 | 90.0±0.2 | 100.0±0.0 | 96.1±0.2 |
| ADMA | 98.1±0.1 | 98.2±0.1 | 91.0±0.2 | 100.0±0.0 | 90.5±0.2 | 100.0±0.0 | 96.3±0.2 |
| TQS | **100.0±0.0** | **100.0±0.0** | **94.2±0.2** | **100.0±0.0** | **94.2±0.2** | **100.0±0.0** | **98.1±0.2** |

## 2. Experiment Details

We introduce additional experiment details in this section, including dataset description, hyperparameter setting and implementation details.

### 2.1. Dataset Description

**Office-31** [7] is a standard domain adaptation dataset of 3 diverse domains, Amazon(A), Webcam(W) and Dslr(D) with 4,652 images in 31 classes.

**Office-Home** [8] is a more complex dataset containing 15,500 images from 4 different domains in 65 classes: Artistic (A) images, Clipart (C), Product (P) images, and Real-world (R) images.

Following previous works [2], for Office-31 and Office-Home, we use all the data in the source domain as labeled data and all the data in the target domain as unlabeled data, which is also used as the unlabeled data pool in our active setting. For evaluation, following previous domain adaptation paper [1], we use the whole target domain as testing data and compute the classification accuracy.

**VisDA-2017** [6] is a simulation-to-real dataset composed of 280K images in 12 classes with two distinct domains: synthetic 3D models and photo-realistic images.

The training split consists of synthetic images and the validation split consists of real images. Following previous works [4], we use the training split as the source domain and the validation split as the target domain since the ground truth labels for the testing split are not available.

### 2.2. Hyperparameter

According to Section 3.4 of the main text, the size of the candidate pool $b'$ and the number of samples $b$ in each selection process are two hyperparameters in the model. For $b$, we use 1% of all target samples. For $b'$, we fix it as 2% of all target samples. Note the $b'$ can be larger than the labeling budget $B$ since we only select a portion of candidates to annotate. We observe that the fixed $b'$ and $b$ can work well across all datasets. The number of times to perform active selection is $B/b$.

For optimizer, we use AdaDelta since it can better adapt to adding new samples during the training process. For optimizer hyperparameters, we perform cross-validation on source data to tune the hyperparameters. For all the datasets, the learning rate is set as 0.1, which is selected in the range $[10^{-4}, 1]$. The batch size is 32 and the number of training epochs is 30.

In Equation (6) of the main text, We can control $\mu$ to decide a soft separation between normal samples and outliers. We can tune $\sigma$ to control the difference between the largest domainness ($D(F(x)) = \mu$) and the smallest one ($D(F(x)) = 0$ or 1). In the experiments, We observe the density distribution of normal samples and outliers, and fix $\mu$ as 0.75 and $\sigma$ as 0.4, which works well for all the datasets.

Table 3. Classification accuracy on **Office-Home** with 10% target samples as the labeling budget for active methods.

| Method | A→C | A→P | A→R | C→A | C→P | C→R | P→A | P→C | P→R | R→A | R→C | R→P | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ResNet | 42.1 | 66.3 | 73.3 | 50.7 | 59.0 | 62.6 | 51.9 | 37.9 | 71.2 | 65.2 | 42.6 | 76.6 | 58.3±0.1 |
| RAN | 61.6 | 80.5 | 80.1 | 61.8 | 76.8 | 73.9 | 64.1 | 60.2 | 78.4 | 72.9 | 62.6 | 84.8 | 71.5±0.5 |
| UCN | 65.5 | 85.0 | 82.6 | 63.7 | 80.9 | 76.0 | 66.7 | 62.1 | 80.3 | 74.6 | 65.2 | 88.6 | 74.3±0.3 |
| QBC | 66.1 | 84.4 | 81.4 | 64.1 | 80.2 | 74.6 | 67.2 | 62.9 | 78.8 | 73.3 | 66.2 | 86.9 | 73.9±0.2 |
| Cluster | 65.2 | 83.1 | 81.0 | 64.0 | 79.6 | 74.2 | 65.1 | 60.5 | 77.8 | 73.0 | 65.2 | 86.1 | 72.9±0.2 |
| AADA | 65.8 | 84.5 | 82.2 | 64.1 | 80.6 | 76.1 | 67.6 | 62.6 | 80.1 | 73.7 | 66.1 | 88.6 | 74.3±0.2 |
| ADMA | 66.5 | 85.4 | 82.8 | 63.8 | 80.9 | 76.3 | 67.7 | 61.6 | 80.9 | 74.3 | 66.8 | 89.7 | 74.7±0.3 |
| TQS | **68.0** | **87.7** | **85.7** | **67.0** | **83.0** | **78.7** | **69.3** | **64.5** | **83.9** | **77.8** | **68.9** | **90.6** | **77.1±0.3** |

Table 4. Classification accuracy on **Office-Home** with 20% target samples as the labeling budget for active methods.

| Method | A→C | A→P | A→R | C→A | C→P | C→R | P→A | P→C | P→R | R→A | R→C | R→P | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ResNet | 42.1 | 66.3 | 73.3 | 50.7 | 59.0 | 62.6 | 51.9 | 37.9 | 71.2 | 65.2 | 42.6 | 76.6 | 58.3±0.1 |
| RAN | 71.0 | 87.0 | 84.2 | 70.2 | 83.1 | 79.9 | 71.7 | 70.2 | 82.5 | 77.0 | 73.1 | 87.9 | 78.2±0.4 |
| UCN | 76.3 | 91.4 | 87.4 | 73.3 | 89.1 | 83.2 | 74.8 | 73.4 | 85.4 | 80.0 | 75.8 | 93.9 | 82.0±0.2 |
| QBC | 77.1 | 90.8 | 85.9 | 73.2 | 88.1 | 81.0 | 75.1 | 74.7 | 83.1 | 77.8 | 76.8 | 91.3 | 81.2±0.2 |
| Cluster | 75.9 | 89.5 | 85.4 | 72.6 | 87.1 | 80.4 | 72.8 | 70.8 | 81.7 | 77.3 | 75.8 | 90.0 | 79.9±0.2 |
| AADA | 76.7 | 90.9 | 86.9 | 74.7 | 88.7 | 83.3 | 75.7 | 74.3 | 85.0 | 78.5 | 76.7 | 93.9 | 82.1±0.2 |
| ADMA | 77.5 | 91.8 | 87.6 | 73.9 | 89.1 | 83.5 | 75.6 | 72.6 | 86.3 | 79.5 | 77.4 | 95.6 | 82.5±0.2 |
| TQS | **79.6** | **94.0** | **91.2** | **77.6** | **92.1** | **87.2** | **78.7** | **77.4** | **90.8** | **85.7** | **79.6** | **97.0** | **85.9±0.2** |

### 2.3. Implementation Details

We implement our algorithm in PyTorch [5]. We use ResNet-50 [3] pre-trained on ImageNet as the backbone network. We run each experiment 3 times to compute the average and the standard deviation. We use PyTorch 1.7, torchvison 0.6 and CUDA 11 libraries.

We use a machine with 32 core CPUs, 256 GB memory and one NVIDIA TITAN RTX. The average training time for each run is about 0.5 hours.

## 3. Results

We provide more experimental results in this section, including classification accuracy on more labeling budget and the sample overlap ratio of different criteria.

### 3.1. Classification Accuracy

In this section, we give more results for three dataset Office-31 [7], Office-Home [8] and VisDA-2017 [6] with different labeling budgets.

We further run experiments on the 6 tasks on the Office-31 dataset and 12 tasks on the Office-Home dataset with 10% and 20% labeling budgets. Table 1 and 2 show results on Office-31 with 10% and 20% target samples as the labeling budget. Table 3 and 4 show results on Office-Home with 10% and 20% as the labeling budget. We can observe that Transferable Query Selection (TQS) consistently outperforms other active learning and active domain adaptation methods with different labeling budgets. The results demonstrate that the proposed TQS works well stably with various labeling budgets, which is an important property for a successful active selection criterion.

Table 5. Classification accuracy on **VisDA-2017** with different percents of target samples as labeling budget for active methods.

| Method | 2% | 5% | 10% | 20% |
|---|---|---|---|---|
| ResNet | | 44.7±0.1 | | |
| RAN | 73.4±0.6 | 78.1±0.6 | 82.1±0.4 | 87.2±0.3 |
| UCN | 76.4±0.4 | 81.3±0.4 | 85.4±0.3 | 90.3±0.2 |
| QBC | 75.8±0.3 | 80.5±0.3 | 84.1±0.2 | 89.6±0.1 |
| Cluster | 75.7±0.3 | 79.8±0.2 | 83.5±0.2 | 89.6±0.1 |
| AADA | 76.1±0.4 | 80.8±0.4 | 84.6±0.3 | 89.7±0.2 |
| ADMA | 76.3±0.4 | 81.4±0.4 | 84.8±0.3 | 90.0±0.1 |
| TQS | **78.3±0.4** | **83.1±0.4** | **87.2±0.3** | **92.0±0.2** |

As for VisDA-2017, the amount of target samples is large but the number of classes is smaller, so the portion of data needed for active domain adaptation is smaller. Therefore, we further report the results with 2% target samples as the labeling budget. We also report 10% and 20% as the labeling budget for a comprehensive performance report.

From Table 5, we can observe that TQS consistently outperforms previous active learning and active domain adapta-

Table 6. Comparison between UDA methods combined with TQS or random selection (RAN) on **Office-31**

| Method | A→D | A→W | D→A | D→W | W→A | W→D | Avg |
|---|---|---|---|---|---|---|---|
| CDAN | 92.9±0.2 | 94.1±0.2 | 71.0±0.2 | 98.6±0.1 | 69.3±0.3 | 100.0±0.0 | 87.7±0.2 |
| CDAN+RAN | 94.1±0.3 | 94.8±0.3 | 79.1±0.3 | 99.1±0.2 | 77.0±0.3 | 100.0±0.0 | 90.7±0.3 |
| CDAN+TQS | 95.2±0.2 | 96.3±0.2 | 84.7±0.3 | 100.0±0.0 | 83.4±0.3 | 100.0±0.0 | 93.3±0.2 |
| AFN | 92.1±0.2 | 90.3±0.2 | 73.4±0.2 | 98.7±0.1 | 71.2±0.2 | 100.0±0.0 | 87.6±0.2 |
| AFN+RAN | 93.4±0.3 | 92.7±0.3 | 81.2±0.4 | 99.3±0.2 | 79.3±0.4 | 100.0±0.0 | 91.0±0.3 |
| AFN+TQS | 94.6±0.2 | 93.7±0.2 | 86.0±0.3 | 99.3±0.2 | 84.7±0.3 | 100.0±0.0 | 93.0±0.2 |
| CAN | 95.0±0.3 | 94.5±0.3 | 78.0±0.4 | 99.1±0.2 | 77.0±0.4 | 99.8±0.1 | 90.6±0.3 |
| CAN+RAN | 95.5±0.3 | 96.7±0.3 | 80.5±0.4 | 99.6±0.1 | 79.3±0.3 | 100.0±0.0 | 91.9±0.3 |
| CAN+TQS | 96.0±0.0 | 97.0±0.0 | 86.2±0.0 | 100.0±0.0 | 86.1±0.0 | 100.0±0.0 | 94.2±0.3 |

tion methods. In particular, even with 2% labeling budget, TQS outperforms ResNet, which indicates the efficacy of the proposed TQS even with an extremely small labeling budget.

### 3.2. Combining with UDA methods

We aim to demonstrate that TQS can improve the performance of UDA methods, which can further indicate that the proposed TQS has wide usage in domain adaptation. We also need to demonstrate that the improvement of UDA+TQS over UDA does not naívely come from labeling some target data, so we also compare with UDA with randomly selected labeled target samples, which shows that which samples to select is also important for domain adaptation.

As shown in Table 6, CAN+TQS steadily outperforms CAN+RAN and CAN, AFN+TQS outperforms AFN+RAN and AFN, and CDAN+TQS outperforms CDAN+RAN and CDAN. While with various unsupervised domain adaptation methods, training with the annotated samples selected by TQS, outperforms with randomly selected annotated samples and with no annotated samples. The observations demonstrate that TQS and UDA methods can cooperate with each other to address the domain adaptation problem. As stated in the theoretical insight, TQS and UDA methods address different challenges in domain adaptation. TQS addresses the difference in optimal labeling function, while UDA methods address the distribution shift. These two kinds of methods are complementary and collaborate to achieve better performance.

### 3.3. Overlap Ratio of three criteria.

To empirically demonstrate that the proposed transferable uncertainty, transferable committee and transferable domainness are complementary, we show the overlap ratio of the candidates selected by each pair of individual criteria and all the three criteria in Table 7. We can observe that the overlap ratio is *far from 100%*, which means the three criteria are not redundant but are complementary. Also, the overlap ratios are not very low, which means the criteria are not contradictory.

Table 7. Overlap Ratio (%) on **Office-31** with 5% labeling budget.

| Task | $TQS_c$-$TQS_u$ | $TQS_c$-$TQS_d$ | $TQS_u$-$TQS_d$ | ALL of 3 |
|---|---|---|---|---|
| Ratio | 70.2 | 63.2 | 66.3 | 60.4 |

## References


[1] Yaroslav Ganin and Victor S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pages 1180–1189, 2015. 2

[2] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(1):2096–2030, 2016. 2

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[4] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1640–1650, 2018. 2

[5] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019. 3

[6] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017. 2, 3

[7] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226, 2010. 2, 3

[8] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, pages 5385–5394, 2017. 2, 3

[9] Han Zhao, Remi Tachet des Combes, Kun Zhang, and Geoffrey J Gordon. On learning invariant representation for domain adaptation. *arXiv preprint arXiv:1901.09453*, 2019. 1