ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10095747

IMPROVING BERT FINE-TUNING VIA STABILIZING CROSS-LAYER MUTUAL INFORMATION

Jicun Li[1,2], *Xingjian Li*[3,6], *Tianyang Wang*[4], *Shi Wang*[1,2*], *Yanan Cao*[5], *Chengzhong Xu*[6], *Dejing Dou*[3]

[1] Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
[3] Big Data Lab, Baidu Research
[4] University of Alabama at Birmingham
[5] Institute of Information Engineering, Chinese Academy of Sciences
[6] State Key Lab of IOTSC, University of Macau

ABSTRACT

Fine-tuning pre-trained language models, such as BERT, has shown enormous success among various NLP tasks. Though simple and effective, the process of fine-tuning has been found unstable, which often leads to unexpected poor performance. To increase stability and generalizability, most existing works resort to maintaining the parameters or representations of pre-trained models during fine-tuning. Nevertheless, very little work explores mining the reliable part of pre-learned information that can help to stabilize fine-tuning. To address this challenge, we introduce a novel solution in which we fine-tune BERT with stabilized cross-layer mutual information. Our method aims to preserve the reliable behaviors of cross-layer information propagation, instead of preserving the information itself, of the pre-trained model. Therefore, our method circumvents the domain conflicts between pre-trained and target tasks. We conduct extensive experiments with popular pre-trained BERT variants on NLP datasets, demonstrating the universal effectiveness and robustness of our method.

Index Terms— Pre-trained Language Model, Fine-tuning Stability, Mutual Information

1. INTRODUCTION

Large scale pre-trained language models, such as BERT, have dominated a wide range of NLP tasks. This facilitates a common practice of fine-tuning the pre-trained BERT for a few epochs to adapt the model to downstream tasks. Although fine-tuning is simple and effective, it remains unclear how pre-learned general knowledge benefits specific tasks. This leads to a practical issue that, a fine-tuning process exhibits severe instability, i.e. fine-tuning on the same dataset with different seeds often leads to quite different results (in terms of accuracy), some of which generalize very poorly.

To solve this problem, a major idea of existing studies is to impose an additional regularizer, to constrain the learned parameters or representations around that of the pre-trained model. For example, Li et al. propose L^2-SP[1], a modified form of weight decay that uses the starting point (SP) as a reference. Mixout[2] maintains the pre-trained weights when performing dropout[3] to suppress the deviation from the pre-trained model. SMART[4] and R3F/R4F[5] employ trust region theory to constrain the representation movements when adapting to the target task. Though these sorts of approaches have been demonstrated very effective on some tasks, there in fact exists a severe risk of negative transfer[6], i.e., information learned from the pre-trained task may not always be helpful for the target task [7, 8, 9]. We argue that, the effectiveness of most existing regularization methods is highly based on data-dependent assumptions, which are not always valid. This consequently limits their scope of applicability.

Our work It follows the aforementioned line of research, focusing on regularization but defining a new regularizer from a different perspective. We aim to preserve a pre-trained model's reliable behaviors of cross-layer information propagation, rather than preserving the information itself. Specifically, for two adjacent layers' sequence representations T_i and T_{i+1}, we encourage their mutual information $I(T_i, T_{i+1}) = H(T_i) - H(T_i|T_{i+1})$ to be stable during fine-tuning. This reference value of mutual information is a prior, suggested by the pre-trained model.

To enable an end-to-end training by gradient descent, we employ Mutual Information Neural Estimator (MINE)[10] to estimate the mutual information. Between each two adjacent layers, a light-weight MINE network is plugged in to monitor and regulate the mutual information. Since MINE involves an extra optimization process, we propose to adopt an efficient

The work was partly done during Jicun Li's internship at Baidu Research. Jicun Li and Xingjian Li contributed equally. Supported by the National Key Research and Development Program of China(No. 2022YFC3302300), Advanced Research Project(No. 7090201050307).

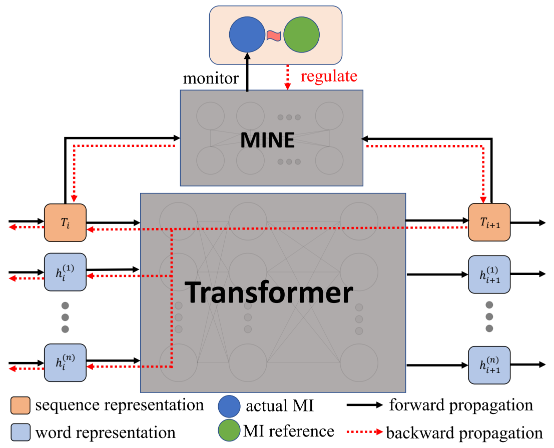

Fig. 1. The pipeline of our approach: fine-tuning via stabilizing cross-layer mutual information (MI).

scheme to perform alternate updates for the MINE network and the backbone network, e.g. BERT. The main idea of our approach is illustrated in Figure 1.

We evaluate the proposed approach with BERT[11] and RoBERTa[12], on both classification (RTE and MRPC) and regression (STS-B) tasks from the popular GLUE benchmark. The results demonstrate that our method consistently improves the vanilla fine-tuning and yields competitive performance compared to the state-of-the-art fine-tuning regularizers, including L^2-SP[1], Mixout[2], SMART[4] and R3F[5]. In addition, we also provide further analyses explaining why our approach delivers good performance.

2. PRELIMINARIES

Problem Definition Unlike standard supervised learning, the fine-tuning process starts from a pre-trained status which holds a parameters-prior obtained from a relevant source task, thus giving a more heuristic initialization than a random one for the model used in a target task. Specifically, we use $f(.;\boldsymbol{\theta})$ to denote the model, of which pre-trained parameters are $\boldsymbol{\theta}^0$. We optimize the model with Stochastic Gradient Descent (SGD) for T iterations. $\boldsymbol{\theta}^t$ denotes the learned parameters at the t-th iteration, and can be generally obtained by

$$\boldsymbol{\theta}^t = \min_{\boldsymbol{\theta}} \mathop{\mathbb{E}}_{\boldsymbol{x}}\{L[f(\boldsymbol{x};\boldsymbol{\theta})]\} + \lambda R(\boldsymbol{\theta}), \tag{1}$$

where L is the empirical loss and R defines the regularization term. λ is the coefficient to balance their effects. Note that in the fine-tuning framework, the regularization term $R(\boldsymbol{\theta})$ can also involve the historical parameters on an optimization trajectory, i.e. $\boldsymbol{\theta}^0, \boldsymbol{\theta}^1, ..., \boldsymbol{\theta}^{t-1}$.

2.1. Regularizers for Fine-tuning BERT

Here we summarize the formats of the state-of-the-art fine-tuning regularizers which are most relevant to our method.

Shrinking towards zero The most common choice is the L^2 penalty imposed on the parameters with the form of $R(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2^2$, also known as weight decay in deep learning. This naive regularizer may lead to catastrophic memory loss of the pre-trained knowledge [1, 2, 4].
Shrinking towards the starting point A straightforward extension of the standard L^2 regularizer is to utilize the starting point as a reference. For example, the L^2-SP regularizer [1] uses the squared Euclidean distance between the learned parameters and the pre-trained parameters as

$$R(\boldsymbol{\theta}) = \|\boldsymbol{\theta} - \boldsymbol{\theta}^0\|^2. \tag{2}$$

Mixout [2] extends the standard form of Dropout [3] in transfer learning context by mixing the pre-trained parameters into Dropout. The proposed $Dropout(\boldsymbol{\theta}, \boldsymbol{\theta}^0)$ acts as an implicit regularizer.
Shrinking towards preceding regions Trust region theory [13, 14] motivates a sort of fine-tuning approaches aiming to suppress aggressive updating [4, 5]. Specifically, SMART [4] employs the Bregman divergence as a regularizer that

$$R(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\boldsymbol{x}}\{KL_s(f(\boldsymbol{x};\boldsymbol{\theta}), f(\boldsymbol{x};\boldsymbol{\theta}^{t-1}))\}, \tag{3}$$

where KL_s denotes the symmetric Kullback–Leibler divergence. Intuitively, such a regularizer prevents a new solution from deviating too much from preceding regions.

Note that SMART [4], as well as FreeLB [15], also introduce an adversarial regularization to smooth the learned model. Aghajanyan et al. discuss its relationship with trust region and propose an adversarial-free implementation R3F [5] as

$$R(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\boldsymbol{x}}\{KL_s(f(\boldsymbol{x};\boldsymbol{\theta}), f(\boldsymbol{x}+\boldsymbol{z};\boldsymbol{\theta}))\}, \tag{4}$$

where z is a random noise subjecting to the Gaussian or Uniform distribution. Hua et al. introduce a similar method that encourages noise stability [16].

3. APPROACH

3.1. Cross-layer Mutual Information

In information theory, mutual information (MI) between two random variables quantifies the amount of information that one variable obtains due to the observation of the other. Let X and Z denote two random variables, respectively. From the perspective of uncertainty, mutual information is defined as

$$I(X, Z) = H(X) - H(X|Z), \tag{5}$$

which can be interpreted as the reduction of the variable X's entropy given another (usually relevant) variable Z observed.

Our proposed *cross-layer mutual information stabilizer* considers the special [CLS] token in BERT. [CLS] is the first token of a BERT input sequence. For classification tasks, every hidden state corresponding to this token is considered to be the aggregate representation of a hidden layer, namely sequence representation T_i in this work. Our method focuses on the mutual information of two adjacent layers' sequence representations, which can be regarded as two relevant random variables.

3.2. Mutual Information Estimator

Dependence of two random variables can also be measured by the distance between their joint probability distribution and the product of their respective marginal distributions. In this sense, mutual information between X and Z can be formulated as

$$I(X, Z) = KL(\mathbb{P}_{XZ} \| \mathbb{P}_X \otimes \mathbb{P}_Z), \tag{6}$$

where KL stands for the standard KL divergance, $\mathbb{P}_{XZ}$ the joint distribution, and $\mathbb{P}_X$/$\mathbb{P}_Z$ the marginal distributions. A recent work MINE [10] converts the KL divergence to its dual representation according to the theorem of *Donsker-Varadhan representation* [17]. By employing a neural network G parameterized by $\phi \in \Phi$ to solve the induced supremum problem, MINE [10] further formulates the estimation of mutual information as

$$I(X, Z) = \sup_{\phi \in \Phi} \mathbb{E}_{\mathbb{P}_{XZ}}[G_\phi] - \log(\mathbb{E}_{\mathbb{P}_X \otimes \mathbb{P}_Z}[e^{G_\phi}]). \tag{7}$$

In practice, a two-layer fully-connected network is employed as G. We use Monte Carlo sampling to approximate the expectation and perform stochastic gradient ascent to approach the supremum. The converged value is regarded as an estimation of the mutual information between X and Z.

3.3. The Overall Framework

Our proposed framework consists of a two-stage procedure. The first stage is to initialize mutual information (MI) estimators and to calculate MI references, using the pre-trained BERT model as a feature extractor. The second stage is to alternately fine-tune the BERT model and the MI estimators.

Initializing MI Estimators Let M denote the number of layers in BERT. We plug in a MINE network between every two adjacent layers. Using T_i to denote the output distribution of the sequence representations at the i-th layer, the MINE network $g(.;\boldsymbol{\phi}_i)$ $(i = 1, 2, ...M-1)$ is responsible for estimating mutual information between the i-th and $(i+1)$-th layer, taking T_i and T_{i+1} as an input, respectively.

To initialize the MI estimators, we freeze the pre-trained BERT and feed forward all training examples at one time. For each training example, we obtain its all M hidden-layer sequence representations, which are then used to optimize Equation 7 until convergence. Afterwards, we have the references of mutual information between every two adjacent layers $\{b_i\}_{i=1}^{M-1}$. Concurrently, the MI estimators are initialized over target examples using the pre-trained BERT as a feature extractor.

Alternately Fine-tuning In this stage, we update both the BERT model and the MI estimators in an end-to-end fashion. The reason why the MI estimators still need fine-tuning lies in that, when deep representations move during BERT fine-tuning, the MI estimators learned in the first stage are no longer accurate. Fortunately, the pre-training process provides a good initialization for the MI estimators, and fine-tuning them will help them catch up the movements of deep representations.

Updating MI estimators When updating the MI estimators, we freeze the BERT model and maximize Equation 7 with stochastic gradient ascent. We empirically find that, updating each MI estimator for one iteration (corresponding to one iteration of BERT fine-tuning) is enough to maintain an accurate approximation.

Updating BERT We freeze the MI estimators while updating BERT. The regularization term used to stabilize cross-layer mutual information is

$$R_{MI}(\boldsymbol{\theta}) = \sum_{i=1}^{M-1} (g(T_i, T_{i+1}; \boldsymbol{\phi}) - b_i). \tag{8}$$

Note that $\{T_i\}_{i=1}^{M}$ are in fact affected by $\boldsymbol{\theta}$, which is omitted in Equation 8 for brevity. With the aforementioned regularizer, the final solution of our approach can be formulated as

$$\boldsymbol{\theta} = \min_{\boldsymbol{\theta}} \mathop{\mathbb{E}}_{\boldsymbol{x}} \{L(f(\boldsymbol{x}; \boldsymbol{\theta})) + \lambda R_{MI}(\boldsymbol{\theta})\}, \tag{9}$$

where L is the standard empirical loss, such as cross entropy in classification. The superscript t in Equation 1 is omitted since our approach does not involve checkpoints in the optimization trajectory. We solve Equation 9 with stochastic gradient descent.

4. EXPERIMENTS

4.1. Experimental Setup

Data To clearly identify which regularization methods are superior, we fine-tune BERT-based models on three small datasets from GLUE [18]: RTE, MRPC and STS-B.

Model We use two BERT-based models: BERT-Large-Uncased [11] and RoBERTa-Large [12] for fine-tuning. In particular, we follow BERT to fine-tune the pre-trained models for 3 epochs, with a learning rate of 2e-5, batch size of 32. AdamW [19] is used as the optimizer. Each experiment is repeated over 20 different random seeds. We report the mean and median of the results obtained from the multiple runs, as well as the standard deviation (std). Our implementation is based on the Huggingface transformers [20].

Baseline Methods Besides the vanilla fine-tuning [11], we compare our method with typical state-of-the-art fine-tuning regularizers, including L^2-SP [1], Mixout [2], SMART [4] and R3F [5].

Table 1. The results of fine-tuning BERT and RoBERTa over 20 random seeds with different regularization methods.

	RTE			MRPC			STS-B		
	mean	std	median	mean	std	median	mean	std	median
BERT	53.94	10.59	47.29	73.93	7.83	68.38	75.94	31.08	89.62
+L^2-SP	53.75	9.27	47.29	75.20	8.64	68.38	70.31	35.07	89.63
+Mixout	50.25	7.26	47.29	73.63	7.53	68.38	88.76	3.08	**89.90**
+SMART	51.82	7.91	47.29	70.98	4.44	68.38	81.31	23.62	89.31
+R3F	51.12	7.88	47.29	79.13	9.10	84.80	82.20	22.29	89.39
+Ours	**67.11**	**6.10**	**69.13**	**85.50**	**2.26**	**85.91**	**89.40**	**0.68**	89.45
RoBERTa	84.80	8.90	86.64	86.35	7.78	89.46	92.15	0.14	92.15
+L^2-SP	87.47	1.47	87.36	86.62	7.91	89.58	87.57	20.73	92.19
+Mixout	78.43	16.03	85.92	89.58	**0.69**	89.58	92.25	0.18	92.23
+SMART	87.11	1.37	87.18	81.67	5.94	84.19	89.38	12.63	92.22
+R3F	87.64	**0.85**	87.55	**90.43**	**0.69**	**90.44**	**92.26**	0.17	**92.31**
+Ours	**87.74**	1.19	**87.73**	89.91	0.75	89.95	92.10	**0.09**	92.09

4.2. Results and Analyses

The summary of experiments is Table 1, our method always outperforms the vanilla fine-tuning, in most cases by a significant margin of the *mean value*. In contrast, the other regularizers sometimes perform worse than the vanilla fine-tuning. Our method yields surprising *mean* and *std value* improvements when fine-tuning BERT. This is because the other methods more or less suffer from failed runs, severely affecting the mean and std values. On the contrary, our method is not subject to failed runs, which eventually benefit the performance.

In most experiments, our method achieves the highest or second-highest *median value*. It is worth noting that, a few completely failed runs yielding very low performance can severely reduce the mean results. In such scenarios, the *median value* is useful to reflect the performance that the most runs are expected to achieve, alleviating the influence of a few outliers. Indicated by the reached higher median values, fine-tuning with our regularizer is more likely to yield a better generalization.

4.3. Effectiveness of MINE

Our regularization method employs MINE for mutual information estimation. To validate that MINE is reliable for estimating cross-layer mutual information of BERT, we compare MINE with the binning method, which is widely adopted for analyzing DNNs. The binning method firstly divides the entire interval of a continuous variable into a number of 50000 uniformly spaced bins, using its maximum and minimum as the bounds. Thus, every sampled continuous value can be converted into a discrete one that belongs to its corresponding bin. Since the binning method is not feasible for high-dimensional variables, we randomly choose five channels and calculate their individual mutual information with binning.

As shown in Figure 2 Left, these mutual information values of different channels exhibit similar trends, though these channels would not be completely independent of each other. The result of MINE also shows a similar trend as that of the binning method.

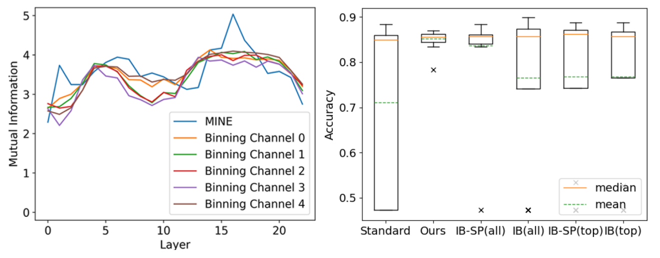

Fig. 2. Left: Cross-layer mutual information estimation results between MINE and binning-based mutual information estimator; Right: Performance distribution of information bottleneck.

4.4. Comparison with Information Bottleneck

The Information Bottleneck (IB) theory [21] is related to our method. Tishby formulates the learning objective as minimizing the IB Lagrangian: $I(X;T) - \beta I(T;Y)$, where the latter term has an effect of minimizing the standard cross entropy. Therefore, IB is equivalent to a regularizer that minimizes the mutual information between a hidden representation T and its corresponding input X. To verify our hypothesis, we compare our method with the original IB and its variant. The original IB minimizes $I(X;T_i)$ for all the $M-1$ layers, while its variant uses the starting point (SP) as a reference (named IB-SP), aiming to minimize $I(X;T_i) - I^0(X;T_i)$, where $I^0(X;T_i)$ is pre-computed with the fixed pre-trained BERT. For each version, in addition to regularizing all the layers, we also evaluate their performance by only regularizing the highest four layers for a comprehensive evaluation.

We illustrate the results in Figure 2 Right, where 'all' refers to regularizing all layers and 'top' refers to only regularizing the highest four layers. We find that the original IB performs significantly worse than our method. However, IB actually improves fine-tuning stability, demonstrating its reasonable assumption. By utilizing the pre-trained BERT as a reference, IB-SP achieves a better overall performance than the original IB. Nevertheless, IB-SP is demonstrated inferior to our method, especially when only regularizing the highest four layers. We speculate this might be because it is rather challenging to build a direct connection between the input and the representations of the higher layers, due to the larger feature dimensions and the inherent optimization difficulties of deep networks.

5. CONCLUSION

In this paper, we explore addressing the issue of instability in BERT fine-tuning. We propose to stabilize the fine-tuning with a regularizer that leverages cross-layer mutual information. With the aid of an appropriate mutual information estimator (i.e. MINE), our method achieves competitive performance on various BERT fine-tuning tasks. Moreover, we interpret how our method relates to the classical Information Bottleneck theory, further demonstrating the rationale of our method.

6. REFERENCES

[1] Xuhong Li, Yves Grandvalet, and Franck Davoine, "Explicit inductive bias for transfer learning with convolutional networks," *Thirty-fifth International Conference on Machine Learning*, 2018.

[2] Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang, "Mixout: Effective regularization to finetune large-scale pretrained language models," in *International Conference on Learning Representations*, 2019.

[3] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[4] Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo 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*, 2020, pp. 2177–2190.

[5] Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta, "Better fine-tuning by reducing representational collapse," in *International Conference on Learning Representations*, 2020.

[6] Lisa Torrey and Jude Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pp. 242–264. IGI global, 2010.

[7] Hao Li, Pratik Chaudhari, Hao Yang, Michael Lam, Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto, "Rethinking the hyperparameters for fine-tuning," in *International Conference on Learning Representations*, 2019.

[8] Xingjian Li, Haoyi Xiong, Haozhe An, Cheng-Zhong Xu, and Dejing Dou, "Rifle: Backpropagation in depth for deep transfer learning through re-initializing the fully-connected layer," in *International Conference on Machine Learning*. PMLR, 2020, pp. 6010–6019.

[9] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi, "Revisiting few-sample bert fine-tuning," in *International Conference on Learning Representations*, 2021.

[10] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm, "Mutual information neural estimation," in *International conference on machine learning*. PMLR, 2018, pp. 531–540.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT (1)*, 2019.

[12] 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.

[13] Razvan Pascanu and Yoshua Bengio, "Revisiting natural gradient for deep networks," *arXiv preprint arXiv:1301.3584*, 2013.

[14] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[15] Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu, "Freelb: Enhanced adversarial training for natural language understanding," in *ICLR*, 2020.

[16] Hang Hua, Xingjian Li, Dejing Dou, Chengzhong Xu, and Jiebo Luo, "Noise stability regularization for improving bert fine-tuning," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 3229–3241.

[17] Monroe D Donsker and SR Srinivasa Varadhan, "Asymptotic evaluation of certain markov process expectations for large time. iv," *Communications on pure and applied mathematics*, vol. 36, no. 2, pp. 183–212, 1983.

[18] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Brussels, Belgium, Nov. 2018, pp. 353–355, Association for Computational Linguistics.

[19] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2018.

[20] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al., "Huggingface's transformers: State-of-the-art natural language processing," *arXiv preprint arXiv:1910.03771*, 2019.

[21] Naftali Tishby, Fernando C Pereira, and William Bialek, "The information bottleneck method," *arXiv preprint physics/0004057*, 2000.