

Building Robust QA System with Few Sample Tuning

Stanford CS224N Default Project RobustQA Track

Liang Yi

Department of Computer Science
Stanford University
chnlyi@stanford.edu

David Yin

Department of Computer Science
Stanford University
yindavid@stanford.edu

Abstract

In this study we aim to modify three sub-optimal general practices of DistilBERT fine-tuning specifically for the question-answering language task, in order to improve both the predicting stability and performance of the model trained by the out-of-domain few samples datasets. We have implemented bias correction for the optimizer, re-initialization of the last transformer block and increase of the training iterations. With smaller sample datasets in the repeated experiment, the major finding is that the F1 score of the model performance has been improved by re-initialization but not by the other two implementations. It is also shown that the stability of finetuned model performance is improved by these implementations even though the improvements are not all statistically significant. In addition, we carry out an additional augmentation step of synonym substitutions for training datasets and show that both F1 and EM (Exact Match) scores are improved in the repeated experiments, with or without last layer re-initialization. Finally, we build a robust ensemble model based on six models that includes data augmentation with and without last layer re-initialization. Our model ranked 2nd place on test leaderboard for F1 as of the time of submission, and achieved 43.096/62.205 (EM)/(F1) on out-of-domain test datasets (March 18, 2021).

1 Key Information

- Mentor: Rui Yan

2 Introduction

Question answering (QA) system is an interdisciplinary field of information retrieval and natural language processing (NLP) that investigates how to build systems capable of automatically answering questions based on context. In the last decade, QA studies in deep learning era have become more advanced and mature by taking advantages of techniques like Word2Vec [1], LSTM [2] and BERT [3]. Even though the pre-trained Contextual Embeddings (PCE) models have often been used as a starter by default in order to obtain state-of-art models for NLP tasks including QA, it has been shown that the seemingly simple transfer learning of BERT-like models are not robust to handling out-of-domain datasets [4][5][6]. In this project we tried several implementations to modify DistilBERT's few-sample finetunings, in order to address the question of model robustness after trained by out-of-domain data.

3 Related Work

It has been shown that adding a step to correct bias of the non-standard Adam optimizer (BERTAdam) implemented in the original BERT helps the model convergence finetuned by datasets of small size [7]. The authors further improved the BERT finetuning stability by the re-initialization of more layers than just the last task-specialized layer of the pre-trained model. Finally, more iterations

were applied to achieve better training stability. The study focused on the finetuning of BERT for four language tasks: natural language inference, paraphrase detection, sentiment classification, and linguistic acceptability. However, it is worth carrying out experiments to answer the question of whether other compact models like DistilBERT finetuned for other language tasks (question and answering) can also achieve the same improvements in stability and performance using these same approaches.

Data augmentation has also shown to be a very promising technique to improve the BERT finetuning model performance [8] [9]. The automatic writing style transformation could be easily achieved by synonym replacement at a level of equal or higher quality [10]. Zhou et al proposed an end-to-end lexical substitution approach based on BERT [11]. It has been shown that the strategy of augmenting the original English training data with back-translated data outperformed the baseline model [12]. Evidently, with the few out-of-domain samples, the finetuning of BERT-like models could be improved by some forms of data augmentation.

Using ensemble methods as learning algorithms can often result in better performance than any single model and has been widely adopted in the field [13]. NLP-based sentiment analysis using ensemble classifiers were demonstrated to outperform the traditional bag-of-words approach [14]. The ensemble approach also boosted strong individual models with significant improvement and benefited a lot from more individual models in neural machine translations [15]. Pranesh et al proposed a state-of-the-art BERT based ensemble language model, QuesBELM, and experimented with various ensemble techniques [16]. Almost all of the top leaderboard models in Question Answering on SQuAD2.0 applied a certain degree of ensembles [17]. An ensemble mechanism of several base models is preferable for the improvement of the few-sample finetuning.

4 Approach

4.1 Baseline

The datasets provided by the default project RobustQA track are relatively large and thus require significant lengths of time for multiple finetuning experiments. To facilitate the comparison of various experimental results in a timely manner, we adopted a sub-sampling routine to create a randomly select sub-sample with 10% of the original dataset. The baseline is obtained by finetuning pre-trained “distilbert-base-uncased” model [18] specialized for the question-answering tasks on just the in-domain datasets (Natural Questions, SQuAD and NewsQA). Because we are going to examine the effect of non-standard Adam optimizer, the baseline model will turn the bias correction switch in AdamW Optimizer off (details in the next section).

4.2 Adam Optimizer

The DistilBert model used a non-standard Adam optimizer. The non-standard Adam optimizer omits a step that facilitates bias correction. This step computes bias-corrected first moment and second raw moment estimates (Figure 1). However a new optimizer named “AdamW” was also implemented with the addition of the step for bias correction, as described in [7]. This version of optimizer has already been adopted in the project’s provided Github repository.

```

 $m_0 \leftarrow 0$  (Initialize 1st moment vector)
 $v_0 \leftarrow 0$  (Initialize 2nd moment vector)
 $t \leftarrow 0$  (Initialize timestep)
while  $\theta_t$  not converged do
   $t \leftarrow t + 1$ 
   $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$  (Get gradients w.r.t. stochastic objective at timestep  $t$ )
   $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$  (Update biased first moment estimate)
   $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$  (Update biased second raw moment estimate)
   $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$  (Compute bias-corrected first moment estimate)
   $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$  (Compute bias-corrected second raw moment estimate)
   $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$  (Update parameters)

```

Figure 1: Bias Correction for Adam Optimizer

4.3 Layer Re-initialization

The “distilbert-base-uncase” model is composed of two blocks: “embedding” and “transformer”. The “transformer” block has 6 transformer sub-blocks, each consisting of “attention”, “sa_layer_norm”, “ffn” and “output_layer_norm”. They are further composed of various combinations of pytorch “Linear”, “LayerNorm” and “Dropout” layers. To re-initialize the parameters, the weights and bias of the “Linear” layers are changed with $\mathcal{N}(0, 0.02)$ and 0 after loading from pre-trained model, respectively. Similarly, the weights and bias of the “LayerNorm” layers are reinitialized with 1 and 0, respectively. Because there are only six transformer sub-blocks in DistilBERT, we decided to only re-initialize the last sub-block to avoid damages to the model quality.

4.4 Number of Iterations

To increase the finetuning iterations, the baseline model is simply adjusted to increase number of epochs to values greater than three. Since these repeated experiments with more epochs execute for a lengthy time, we have two experiments that increased it only to six epochs, with and without re-initialization of the last sub-block of the transformer.

4.5 Data Augmentation

The major data augmentation methods for NLP projects include lexical substitution, back translation, text surface transformation, instance crossover augmentation, syntax-tree manipulation, MixUp for text, generative methods, etc. We adopted the thesaurus-based synonym substitution by nlpaug [19]. It is optional to be extended to word embedding or masked language model based substitution as well. The percentage of the substitution will be examined to find the optimal value for the model. Meanwhile the targeted datasets for the augmentation operations will provide options to determine which domain or domains of the training datasets contribute the most to the performance gain. Due to the nature of the few-sample out-of-domain datasets, the augmentation should be focused towards them. This might also compensate the lack of out-of-domain data relative to the in-domain so that they become more balanced relative to the in-domain datasets.

4.6 Ensemble

Deep networks like pre-trained BERT-like models are further trained (finetuned) on task specific datasets. They can become highly sensitive to initial conditions, which can be corrected by ensemble models. In addition to reducing the variance in the prediction, the ensemble can also result in better predictions than any single best model. There are three major elements in the ensemble method: Training Data, Ensemble Models, and Combinations. For this project we simple adopt the unweighted average ensemble of m base models for the QA start and end words’ probabilities:

$$\begin{aligned} start_prob(i) &= \frac{1}{m} \sum_{k=1}^m P_k(i) \\ end_prob(j) &= \frac{1}{m} \sum_{k=1}^m P_k(j) \end{aligned}$$

5 Experiments

5.1 Data

The data provided are three in-domain reading comprehension datasets (Natural Questions, SQuAD and NewsQA), and three out-of-domain datasets (RelationExtraction, DuoRC, RACE). The questions were crowdsourced for SQuAD, NewsQA, and DuoRC, obtained from search logs for Natural Questions, and from teachers in RACE and synthetic for RelationExtraction. The passages were from a mix of Wikipedia, news articles and exam data. In all training and validation data, every passage is paired with a corresponding question and answer. The in-domain data are divided such that there are 50,000 examples in training for each, and the rest (10,507 for SQuAD, 4,212 for NewsQA, and 12,386 for Natural Questions) for validation. In the out-of-domain datasets, there are 127 examples

each for training, and similar amounts for validation. There will be test examples (ranging from 400 to 2000+) for these out-of-domain datasets where the answers are not provided.

The in-domain and out-of-domain datasets are split into training and validation data folders. Each of the training datasets are randomly sub-sampled at 10% to obtain the corresponding smaller datasets for experiments. For the preliminary experiments to compare training stability and performance, the sub-sampled training datasets are used along with full in-domain validation datasets, and are compared against each other using out-of-domain validation datasets. For the final dev and test leaderboard submission, the full original training datasets and all validation datasets are used.

We will refer to the term all-domain as including both in-domain and out-of-domain for training or validation purposes.

5.2 Evaluation method

The performances of all the models are measured via two metrics: Exact Match (EM) score and F1 score. The maximum F1 and EM scores across the three human-provided answers will be averaged to provide final scores. As planned in the proposal, the stability of the finetuning is evaluated by repeated experiments.

5.3 Experimental details

5.3.1 Small Sample Finetuning of Selected Models

To facilitate carrying out multiple experiments in a timely manner, we have carried out five finetuning experiments using small sub-samples of all-domain training data and one evaluation using the complete out-of-domain validation datasets. The experiments are based on the enhancements and modifications referred in the Approach section. This experiment allows us to obtain a preliminary model that outperforms the baseline model for an intermediate leaderboard submission.

5.3.2 Data Augmentation and Hyperparameter Search

We plan to augment proportions of the dataset with synonyms of the original words, as described in [20]. We examine the effect of augmentations for all training datasets including in-domain and out-of-domain vs out-of-domain only. To do this, we run a hyperparameter search over various experiments of the data augmentation method on the small training dataset. Based on the F1 score, we choose the best insertion rate of the synonyms and percentage substituted to include in our final experiments and subsequent model selection.

5.3.3 Small Sample Finetuning Stability Analysis

We have carried out 6 sets of finetuning experiments, each repeated 12 times using small all-domain training and small in-domain validation datasets in finetuning, and tested by small out-of domain validation dataset. The experiment’s details are listed in Table 1. Note that “_aug_ood” designates augmentation for out-of-domain.

Experiment Name	Adam Bias	Layers Re-init	Epochs	Augmentation
ad_base_adam	yes	0	3	no
ad_base	no	0	3	no
ad_base_reinit	no	1	3	no
ad_base_reinit_ep6	no	1	6	no
ad_base_aug_ood	no	0	3	yes
ad_base_reinit_aug_ood	no	1	3	yes

Table 1: Experimental Details

5.3.4 Ensembling and Model Selection

Based on the results of the finetuning stability analysis, we make a boxplot of the F1 and EM scores for each experiment type and conduct an F-test for variability across the models. From this, we make an informed decision on which model or models to use for a final finetuning on the full train dataset. We will then make an ensemble of 6 such models to produce our final predictions for submission.

5.4 Results

5.4.1 Small Sample Finetuning of Selected Models

The results of the experiments are summarized in Table 2. Compared with the baseline model trained on in-domain datasets only, there is a sizable improvement of performance gain in the model trained on all-domain datasets. Among other models trained on all-domain however, the F1 scores are hardly different (or slightly lower), with EM scores being lower. Thus, the modifications of re-initialization appeared to hurt the models’ performances, even though a faster drop of loss was observed during the training (Figure 2). The final evaluation using complete out-of-domain validation datasets has been submitted to the dev leaderboard with the scores of F1: 49.13 and EM: 33.77.

Experiment Name	F1	EM	Used for Final Training
Baseline	37.81	21.47	N
all_domain_baseline	40.65	26.70	Y
all_domain_reinit	40.02	23.56	N
ad_ep6	40.65	26.70	N
ad_reinit_ep6	40.02	23.56	N

Table 2: Experimental Results

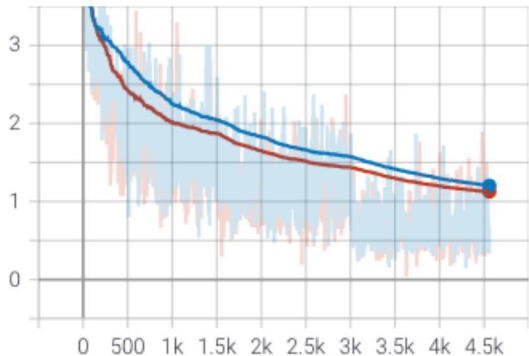


Figure 2: Training Loss Change in Baseline (Blue) and Re-initialization (Red)

5.4.2 Data Augmentation and Hyperparameter Search

We have examined the effect of data augmentation for all training datasets including in-domain and out-of-domain vs out-of-domain only. As shown in Figure 3A, the result implied that including in-domain data augmentation did not help, and even appeared to have hurt the EM performance. In addition we tested how many times of synonym-substituted out-of-domain data produced the optimal result, and discovered that 8 times is the best (Figure 3B). Finally we tested the optimal percentage that should be used for synonym substituted data insertion and found it to be 10 to 20% (Figure 3C). Therefore in all the repeated experiments that used “_aug_ood”, the insertion rate was chosen to be 8 times and the substitution percentage to be 20%.

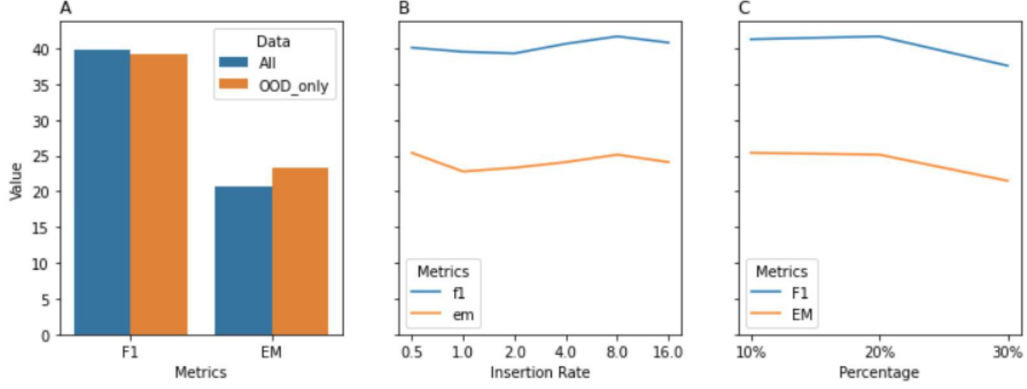


Figure 3: Synonym Substitution Experiments

5.4.3 Small Sample Finetuning Stability and Performance Analysis

The two plots in Figure 4 show the variation of the F1 and EM scores for these 6 different models based on 12 runs each. Interestingly, we did not find the approaches mentioned by Zhang et al [7] to improve finetuning stability on the small train dataset. There are only 2 treatments (ad_base_adam vs ad_base_reinit and ad_base_adam vs ad_base_reinit_ep6) that showed statistical significance for the EM score variance difference by F-tests (Table 3), and the models with these changes had variations in performance that are not any less than the model without applying these changes (ad-base). However in Figure 4, pairwise comparisons for many experimental conditions show that performance increased with the inclusion of our approaches, especially for the ones with data augmentations. Since the top two models have the lower quartile results to be as good or better than the upper quartile results of the other 4 models, and that there is no noticeable difference in performance of the top two models between themselves, we decide to use these two models for the final ensembling.

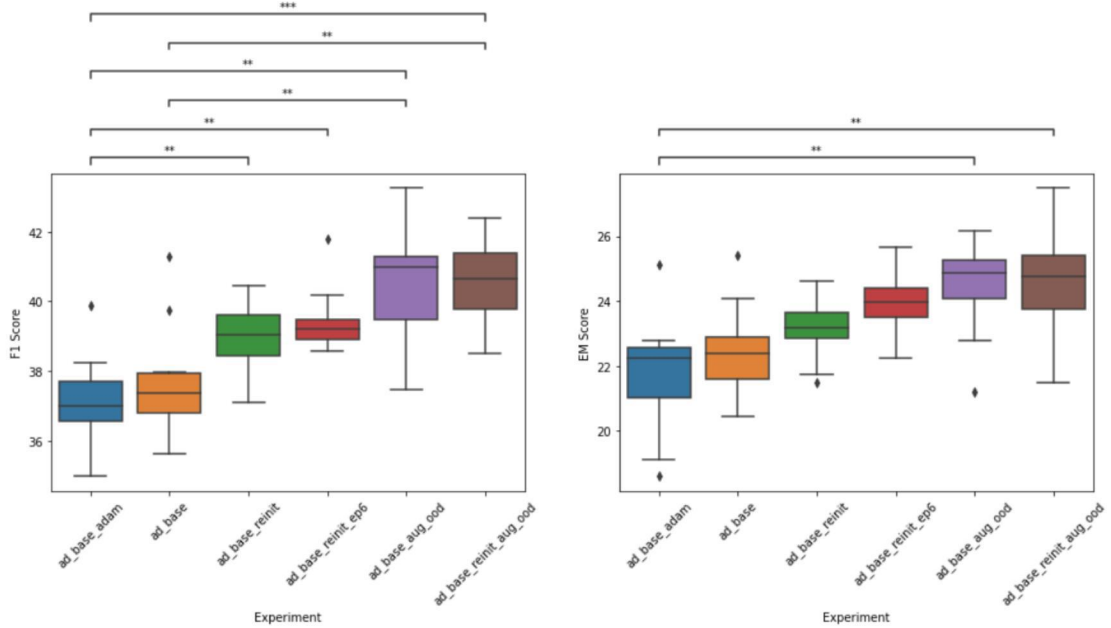


Figure 4: Boxplot of F1 and EM Scores for Performance and Stability Experiment (p-value annotation legend: **: $1.00e-03 < p \leq 1.00e-02$; ***: $1.00e-04 < p \leq 1.00e-03$)

Comparison	p-value (F1)	p-value (EM)
ad_base_adam vs ad_base	0.688234	0.236991
ad_base_adam vs ad_base_reinit	0.204847	0.020543
ad_base_adam vs ad_base_reinit_ep6	0.096913	0.030497
ad_base_adam vs ad_base_aug_ood	0.737985	0.254149
ad_base_adam vs ad_base_reinit_aug_ood	0.376492	0.375588

Table 3: F-tests for Equalities of Variance of F1 and EM Scores

5.4.4 Ensembling and Model Selection

By applying the top two models chosen from the stability analysis, we did six runs of finetuning using the full all-domain train datasets and all-domain validation datasets for each model. The models are evaluated by out-of-domain val datasets and the results are summarized in Figure 5. The resulting 12 models are used as base models for ensemble to make the final predictions on the out-of-domain test dataset. The start and end probabilities for each observation for the 12 models are ensembled according to our aforementioned approach to generate our final submission. Our final test submission has the scores of 43.096/62.205 (EM)/(F1) and the F1 score is temporarily ranked 2nd in the leaderboard (March 18, 2021).

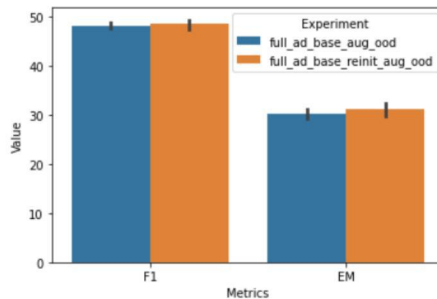


Figure 5: Evaluation for Data Augmentation with and without Re-initialization

6 Analysis

Our results show that the three modifications adopted from Zhang et al [7] do not stabilize the few sample finetuning. This is probably due to the relative small-sized DistilBert model compared with BERT, and the intrinsic differences of the models. It may be worth trying expanded experiments to further identify other sub-optimal practices in DistilBert and other language tasks.

Overall, the result of the final model has been outstanding as indicated by the model’s performance on the final test set shown on the leaderboard. However, there are some weaknesses and limitations in our design that could have been improved upon.

During experimentation, we decided to use the in-domain validation set for validation so that we would be able to leave the full out-of-domain validation set for evaluation. This may have led to the experimented models biased toward answering in-domain questions better whereas not necessarily learning how to answer out-of-domain datasets as best as it could. Due to this, the experimental results may have biased our model selection. A proper solution may be to split the out-of-domain validation by a proportion of 70/30 for example, and use the first part for validation set during finetuning and the second part for evaluation.

Another limitation is with the size of our finetuning set during experimentation. We created a one-time subsample of 10% of the full train dataset in the beginning which was used for all subsequent experimentation finetunings. This may have limited our model selection as the experiments could only learn nuances of the subsample rather than seeing the entire train dataset. For the validation sets in our second stability analysis experimentation, we also used a 10% subsample of the full

validation data. Similarly, this may have biased the model parameters and may also have resulted in hyperparameters chosen to better predict on the small validation dataset instead of on the full validation dataset, which may be more representative of the hidden test set.

7 Conclusion

Overall, our project has shown that the bias correction of the non-standard Adam optimizer led to a general better performance, although with no change in performance variability. Similarly, the re-initialization of the last layer, along with data augmentation through the use of synonym replication have also shown to improve the BERT finetuning model performance [8] [9]. Furthermore, ensembling of the top selected models from experimentation appear to have resulted in a strong leaderboard performance as expected. The efforts of this project clearly show that it is very promising for the compact models like DistilBERT to be optimized for few sample finetuning and for improvement of robustness. Future works for the model could include trying other data augmentation methods such as syntax-tree manipulation, checking the model's robustness on more unseen language tasks, and testing new ensemble mechanisms.

References

- [1] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.
- [2] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks, 2014.
- [3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
- [4] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems, 2017.
- [5] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data, 2018.
- [6] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics.
- [7] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. Revisiting few-sample bert fine-tuning. *arXiv:2006.05987*, 2020.
- [8] Shayne Longpre, Yi Lu and Zhucheng Tu, and Chris DuBois. An exploration of data augmentation and sampling techniques for domain-agnostic question answering. *arxiv.org/pdf/1912.02145.pdf*, 2019.
- [9] Wei Yang, Yuqing Xie, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. Data augmentation for bert fine-tuning in open-domain question answering. *arxiv.org/pdf/1904.06652.pdf*, 2019.
- [10] F. Khosmood and R. Levinson. Automatic synonym and phrase replacement show promise for style transformation. In *2010 Ninth International Conference on Machine Learning and Applications*, pages 958–961, 2010.
- [11] Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei, and Ming Zhou. BERT-based lexical substitution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3368–3373, Florence, Italy, July 2019. Association for Computational Linguistics.
- [12] Mihaela Bornea, Lin Pan, Sara Rosenthal, Radu Florian, and Avirup Sil. Multilingual transfer learning for qa using translation as data augmentation, 2020.
- [13] Thomas G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.
- [14] M. Kanakaraj and R. M. R. Guddeti. Nlp based sentiment analysis on twitter data using ensemble classifiers. In *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, pages 1–5, 2015.
- [15] Yiren Wang, Lijun Wu, Yingce Xia, Tao Qin, ChengXiang Zhai, and Tie-Yan Liu. Transductive ensemble learning for neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):6291–6298, Apr. 2020.
- [16] R. R. Pranesh, A. Shekhar, and S. Pallavi. Quesbelm: A bert based ensemble language model for natural questions. In *2020 5th International Conference on Computing, Communication and Security (ICCCS)*, pages 1–5, 2020.
- [17] Question Answering on SQuAD2.0, url = <https://paperswithcode.com/sota/question-answering-on-squad20>.
- [18] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv:1910.01108*, 2020.
- [19] Edward Ma. Nlp augmentation. <https://github.com/makcedward/nlpaug>, 2019.
- [20] W. Zhou. Ensemble bert with data augmentation and linguistic knowledge on squad 2.0, 2019.