Know When To Abstain: Calibrating Question Answering System under Domain Shift

Stanford CS224N Custom Project
Mentor: Robin Jia

Wanze Xie
Department of Computer Science
Stanford University
wanzexie@stanford.edu

Ruocheng Wang
Department of Computer Science
Stanford University
rcwang@stanford.edu

Abstract

No one can be an expert on everything. The same is true for Question Answering (QA) systems. During the interaction with users, it is essential for QA systems to correctly understand when it should refrain from giving the answer when it is likely to err. This motivates the problem of confidence modeling: if we can know when a model is unconfident, we can choose to reject its answer, making the model more robust and practical in real-world settings. In this project, we focus on confidence modeling of QA systems under domain shift. Our experiments show that when facing out-of-domain (OOD) questions, QA models can often be over-confident (i.e.: high softmax probability, which is served as our baseline) on incorrect answers. To tackle this problem, we propose a systematic approach to calibrate the model by augmenting it with a calibrator trained on a small subset of out-of-domain examples. And experiments have demonstrated that our calibrator has better modeling of the confidence score, mitigating the issue of out-of-domain over-confidence.

1 Introduction

Language models often tend to have weaker performances on test examples with domain knowledge that are not covered by the training data [1, 2], a situation known as Domain Shift, and such examples are known as out-of-domain (ODD) examples. We observe that Question-Answering (QA) models also suffer from this problem by outputting wrong answers with high confidence. This motivates the problem of confidence modeling, where the QA model is calibrated to have reasonable confidence estimation under domain shift and abstain from giving the answer when it is like to err.

We highlight two distinctions of our setting from problems like transfer learning [3] or out-of-domain detection [4]: (1) We assume little access to a lot of OOD data, and also no access to some other OOD domains. This is closer to the real scenario since it is unlikely to exhaust all domains that user input may come from. (2) We are more interested in calibrating the confidence of the model, rather than actually improving model's performance on them, since the eventual goal is to allow QA model make better decision on when to abstain.

In this paper, we create a QA model by finetuning the pretrained Bidirectional Encoder Representations from Transformers [5] (BERT) model on the Stanford Question Answering Dataset (SQuAD 1.1) and valuate the model's confidence on OOD datasets from MRQA 2019 shared task [6]. We first examine the overconfidence problem of MaxProb, which directly uses the maximum softmax probability of the model as the confidence score. Then we propose *calibrator*, a binary classifier to predict if the QA model is correct or incorrect given the test input, whose confidence is used to decide abstention. We conducted extensive experiments to explore the design of the calibrator, and our results show the following:

- Calibrator with simple features like the QA model softmax probabilities and input length already outperform MaxProb model, and adding features of attentions can significantly enhance calibrator's confidence modeling on the original QA model.
- Calibrator performs the best when trained with a mixture of in-domain data and known OOD data, rather than trained with only in-domain data or only known OOD data.

- Even though the objective of the calibrator is not the same as the one for out-of-domain detection, the task of OOD detection can significantly improves the performance of calibration under domain shift.

2 Related Work

Confidence Modeling Modeling predictive confidence has been crucial for deployment of machine learning models, which has attracted a lot of interest in general deep learning community. A common approach tries to exert a prior distribution over the parameters of the model, and transform the prediction to approximate Bayesian inference [7, 8]. Another approach leverages the idea of ensemble, accumulating predictions from multiple models to estimate the confidence score[9]. But resulting frameworks with such approaches are often expensive in computation and storage since they require multiple inference passes and multiple instances of models. The third approach treats confidence modeling as another prediction problem, utilizing internal representations of the model and other evidence as features to predict the confidence score. In [10], the softmax probability is directly used as the confidence score. On the other hand, [11] leverages the embeddings of intermediate layer for confidence estimation. In this paper we focuses on the third approach, which introduces less computation overhead and thus more useful in real world settings.

Confidence Modeling in NLP Confidence modeling has been studied for tasks like semantic parsing[12], machine translation[13], and question answering[14]. [12] and [13] both utilize the characteristic sequence-to-sequence architecture, designing certain heuristics to estimate the confidence, but focus on different tasks from ours. [14] is most similar with our task, but adopts a more general approach that feeds intermediate representations to a neural network for confidence modeling. It does not take into account the architecture specialty of question answering, like the widely used attention mechanism. Furthermore, to the best of our knowledge, current works concentrate on in-domain setting. How confidence estimation will perform on out-of-domain data is unexplored. In this project, we investigate how common confidence modeling technique will behave under domain shift, and propose a calibration framework that maintains performance for this setting.

Prediction with Abstain Many of the previous work [12, 15, 16] has shown that predictions with options to abstain, also known as selective prediction, has been explored in many different machine learning areas. Specifically, [12] proposed the method of using a calibrator to get a better confidence modeling for semantic parsing models, which is a similar approach we discuss in this paper. Moreover, selective prediction under domain shift, which is closer to our setting, is a less explored area especially in for NLP tasks. Some work [17] in NLP discussed the problem of prediction under domain shift, but does not focuses on confidence modeling on OOD dataset. In other fields like medical applications, [18] conducted experiments of training on certain distribution of patients and tested on the other, but in a setting different from this project as well.

3 Approach

3.1 Domain Shift Settings

In our problem setting, we are interested in the confidence modeling of the QA model on the test input that comes from a mixture of in-domain and OOD distributions, with prior exposure to some data from a known OOD distribution.

Formally, we denote the in-domain data, i.e. the training data's distribution, as p_{source}, in our case being the SQuAD 1.1 data, the OOD distribution which we have a few accessible data as q_{known}, and unknown OOD distribution as q_{unk}. Our objective is to achieve good estimation of confidence when QA models are tested on a mix of p_{source} and q_{unk}, as in common real-world scenarios.

3.2 Baseline

We use MaxProb as our baseline, which considers an extractive QA model's softmax probability $p(\hat{y})$ of the predicted answer span $\hat{y}$ as the confidence score, representing the model's certainty level about the answer. MaxProb is known [10, 19] to be a simple yet effective method for out-of-domain detection and confidence modeling tasks, and can be extracted directly from any QA model that assigns probabilities to span to predict the answer. This can be seen as a QA model's inherent confidence score even if the model is not explicitly trained to model confidence.

The MaxProb model will be implemented in two ways. We first explore the performance of MaxProb on **original QA model** (i.e. the pretrained BERT model finetuned on p_{source}), and then we explore its performance on

retuned QA model (i.e. the pretrained BERT model finetuned on p_{source} + q_{known}). In our experiments, we compare both MaxProb models' performance with the calibrator.

3.3 Calibrator

Since the nature of the calibration task is to understand if the confidence score matches the correctness of the prediction outcome, our calibrator can be designed as any binary classifier. In this study, we use the gradient tree boosting model XGBoost [20], primarily because deep and large model architecture does not apply because our setting constraints that the data from q_{known} is limited.

During training, we try to optimize the following binary cross-entropy:

$$J(x,z) = -\frac{1}{N}\sum_{i=1}^{N} z_i \cdot \log(c(x_i)) + (1-z_i)\cdot \log(1-c(x_i))$$

For a prediction of the QA model, z_i indicates whether the model gives the correct answer, $c(x)$ is the confidence score predicted by the calibrator, with x being the features for calibration.

In our experiment, we choose the following type of features for calibration:

Question and Answer Length Number of tokens in the question and answer generated by the tokenizer of the QA model.

Top 5 softmax probabilities Rather than only taking the maximum softmax probability(MaxProb) into account, we leverage the probabilities of top 5 candidates as features.

Variance of top 5 softmax probabilities Intuitively, the variance of the top 5 softmax probabilities can reflect how uncertain the model is choosing between top 5 candidate answers.

Attention scores As attention mechanism is widely used in modern QA architectures, as well as in BERT, we want to see whether extracting features from attentions can help calibrator model the confidence score. The intuition rests in that QA model tends to output wrong answers when some words from the question texts are not attended properly. Specifically, for a prediction of a QA model that leverages self-attention mechanism [21], we derive a types of attention feature which is computed as follows:

$$f_{att} = \min_{i\in S_q}\max_{j\in S_c} a_{ji}$$

where a_{ji} is the self-attention score from j to i, S_c, S_q are the sets of indices of tokens in the context and question.[1] In the formula, we firstly compute how much each token in the question is attended by context tokens, and then we take the mininum over all the question tokens.

The calibrator should be trained with a small set of OOD data from q_{known} and a small set of original in-domain data p_{source}, and be tested on a mixture of unseen data from p_{source} and q_{unk}. Section 4.1 describes this strategy in detail. We aims to prove that with a limited amount of out-of-domain data, the calibrator can be tuned to calibrate the model's confidence and improve its robustness.

4 Experiments

4.1 Data

We leverage pre-processed datasets from MRQA 2019 shared task [6], from which we use SQuAD 1.1 [22] as the in-domain dataset p_{source}, and HotpotQA [23], NewsQA [24], TriviaQA [25], SearchQA [26], and NaturalQuestions [27] as OOD datasets. The six datasets are all for the task of English-language extractive question answering, making them useful for the domain shift setting. The input and output format of the data to the QA model is the same as the one described in original BERT paper [5].

4.2 Evaluation method

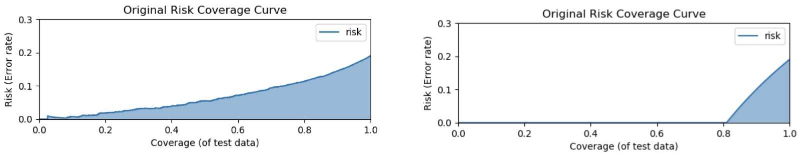
We adopt a total of four evaluation (two quantitative and two qualitative) methods to analyze the performance of our confidence modeling techniques.

[1] Here we calculate the attention features specifically for BERT model where context and question are concatenated as input. Similar approach can be used in other self-attention models

Quantitative Metrics The risk-coverage (RC) curve [28, 16] is a measure of the trade-off between the coverage (the proportion of test data encountered), and the risk (the error rate under this coverage). Since each prediction comes with a confidence score, given a list of prediction correctness Z paired up with the confidence scores C, we sort C in reverse order to obtain sorted C', and its corresponding correctness Z'. Note that the correctness is computed based on Exact Match (EM) as described in [22]. The RC curve is then obtained by computing the risk of the coverage from the beginning of Z' (most confident) to the end (least confident). In particular, these metrics evaluate the *relative order* of the confidence score, which means that we want wrong answers have lower confidence score than the correct ones, ignoring their absolute values. An example of the RC curve and the best possible AURC is shown as in Figure 1. We leverage this metric in two ways:

(1) We use **Area Under the RC curve (AURC)** as a metric for the performance of the confidence modeling technique. A well-calibrated confidence score should have a low AURC and the best possible AURC is constrained by the model's test error and the number of test examples.

(2) We compute **Coverage at Accuracy (Cov@Acc)** by checking the coverage proportion when the RC curve goes beyond the risk defined by the given accuracy, and a well-calibrated model should have a coverage as close as possible to its test accuracy.



(a)

(b)

Figure 1: (a) is an example of the risk coverage curve obtained from testing a QA model on in domain data. The AURC is the filled blue area and the test error of the model is the risk value when the coverage is 1. (b) shows the best possible RC curve under same model, and the filled area represents the best possible AURC.

Qualitative Metrics To compensate for the fact that our quantitative metrics only focuses on the relative order of the confidence score, we also utilized two qualitative methods to visualize the significance of the calibrated confidence score. The examples of both methods are demonstrated in Figure 2.

(1) We visualize **Distribution Plot**, which demonstrates the confidence distribution between in-domain and OOD data for MaxProb and calibrator. For a well-calibrated model, most of its output should have low confidence scores on OOD input, and high confidence scores on in-domain input.

(2) We also draw **Calibration Plot** as a referential qualitative metric, which visualize the effect of our calibration result. The calibration plot shows if the confidence score matches the answer's actual probability of correctness.

4.3 Experimental details

In our experiments, each time we pick one of the OOD dataset as q_{known} and one of the other OOD datasets as q_{unk}. The test set contains a total of 8,000 samples with half from the test set of p_{source} and half from q_{unk}. The training data consists of the training set from p_{source} and 2,000 sampled from q_{known}. The different ways we utilize the training data are described below. All, if not otherwise indicated, are sampled in a random measure.

QA model We used the pretrained BERT-base [5] uncased model (12 attention heads, 12 layers, hidden states dimension 768) from [29], and trained it on SQuAD 1.1 for 2 epochs, with learning rate $3 \cdot 10^{-5}$. This model architecture is used for both the original QA model and the retuned QA model.

The training data for the original QA model are the in-domain training set from p_{source}, and the training data for the retuned QA model further include the 2,000 samples from q_{known}. Note that q_{known} may differ in different experimental settings.

Calibrator model The hyperparameters of the gradient boosting tree are achieved via a 5-fold grid search over{max_depth: [2, 10] with step 1, n_estimators: [60, 220] with step 40, learning rate in 0.1, 0.01, 0.05}. The final parameters are max_depth: 2, n_estimators: 60, learning rate: 0.05}. For the attention features, we finally leverage 12 attention features from the 12 heads of BERT's first layer. So the total number of features is 20.

The training set for calibrators contains 2000 samples from p_{source} and 2000 samples from q_{known}, with its original QA model finetuned on the entire training set of p_{source}. This creates 20 different experiment settings, excluding settings where same OOD dataset is used during training and testing.

4.4 Main Results

4.4.1 Original QA Model

From Table 1 we can see that BERT fine-tuned only on SQuAD has favorable performance on SQuAD while performs poor on OOD data, especially on SearchQA. This demonstrates the performance drop of the QA model on OOD data and signals the over-confidence problem as shown in Figure 2.

	SQuAD	HotpotQA	NewsQA	SearchQA	TriviaQA	NaturalQuestions
EM	81.31	44.47	39.27	17.86	49.90	42.22

Table 1: BERT's exact-match accuracy on the six datasets

4.4.2 Calibrator and MaxProb

We first quantitatively measure the performance of our calibrator by computing the AURC of risk-coverage curve for MaxProb and the calibrator, as shown in Table 2, we can see that training on a small portion of q_{known} data will greatly improves the performance of confidence modeling on the q_{known} distribution. A salient observation is that training on q_{known} can often improve the confidence modeling on any q_{unk} dataset that both the model and calibrator have never seen before.

Trained on ↓\Test on →	**HotpotQA**	**NewsQA**	**SearchQA**	**TriviaQA**	**NQ**
(MaxProb)	23.76	19.71	22.77	18.38	16.65
HotpotQA	19.12	18.37	21.27	15.37	16.08
NewsQA	19.85	17.27	21.28	15.14	15.81
SearchQA	20.17	18.16	20.82	15.00	15.53
TriviaQA	19.82	17.66	20.87	14.50	15.48
NaturalQuestions	22.88	18.67	21.10	16.17	15.30
Best Possible	8.00	9.24	15.42	6.78	8.54

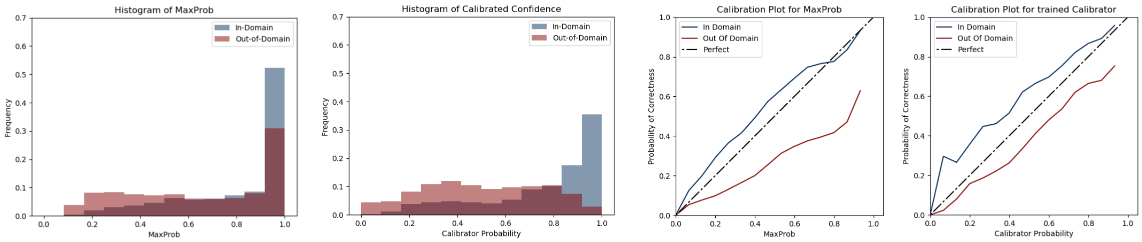
Table 2: AURC of MaxProb and Calibrator, we can see that q_{known} data can help calibrate the predictions on q_{unk} in all situations. Note that in every experiment, there are half of the examples in train and test set coming from SQuAD

Next, we compute our second metrics Coverage at Accuracy by setting accuracy to 80% and also compare against the original MaxProb, as shown in Figure 3. This table reflects the performance of the model when we want to keep the total error rate low to a certain acceptable accuracy threshold. Even though it is expected that the diagonal has generally a better result since it is tested on the same distribution as q_{known}, it is a salient to note that significant improvements of Cov@Acc are identified in every q_{unk} distribution, no matter which OOD dataset is chosen as q_{known}, bringing the score closer to the best possible result.

Trained on ↓\Test on →	**HotpotQA**	**NewsQA**	**SearchQA**	**TriviaQA**	**NQ**
(MaxProb)	35.35	51.95	46.91	55.80	61.20
HotPotQA	49.76	57.46	51.20	65.40	64.60
NewsQA	48.73	59.79	51.83	66.03	65.96
SearchQA	47.54	58.14	52.14	67.10	65.39
TriviaQA	49.05	59.26	52.45	67.71	66.64
NaturalQuestions	43.79	56.35	51.78	65.95	66.51
Best Possible	78.45	75.28	62.35	81.89	77.03

Table 3: Cov@Acc 80% of MaxProb and Calibrator.

Lastly, we investigate the distribution of MaxProb and calibrator confidence on both in-domain and OOD dataset. As shown in Figure 2(a), MaxProb frequently outputs high score either on in-domain or OOD data. However, the red curve in Figure 2(c) indicates that MaxProb is being over-confident on OOD input, which explains the performance drop on OOD. After the calibrator is trained, we generate Figure 2(b) and (d), which indicated that the over-confidence problem is greatly mitigated, since the high confidence on OOD samples has been redistributed to lower confidence intervals, and the calibration curve for OOD data is drawn closer to the perfect calibration curve.[2]



(a) (b) (c) (d)

Figure 2: Comparison between MaxProb and Calibrator using the Distribution Plot (a) (b) and Calibration Plot (c) (d). Note that in the Calibration Plot, if a line fits closer to the dotted line, its corresponding confidence score is a better representation of the actual probability of correctness of its prediction.

4.4.3 MaxProb on Retuned QA Model

As our calibrator achieves better performance with 2000 SQuAD and 2000 OOD examples, it is natural to ask whether simply retune MaxProb model with q_{known} examples can perform better as well. We thus finetuned the pretrained BERT-base uncased model on p_{source} and 2000 examples from different q_{known} and get 5 models, which we note as the retuned models for MaxPorb. We tested this 5 models on respective q_{unk} OOD examples and obtain result in this 20 experiments settings as shown in Table 4.

Trained with ↓\Test on →	**SQuAD**	**HotpotQA**	**NewsQA**	**SearchQA**	**TriviaQA**	**NQ**
Original	81.31	44.47	39.27	17.86	49.90	42.22
HotPotQA	80.20	52.48	39.75	24.45	48.25	45.60
NewsQA	80.73	43.65	43.78	22.90	50.63	46.70
SearchQA	80.63	43.08	39.88	53.80	54.23	45.80
TriviaQA	80.18	46.25	38.48	21.60	47.60	38.98
NaturalQuestions	80.03	42.38	40.45	24.98	50.95	56.28

Table 4: Exact accuracy of BERT which has access to OOD data. Original SQuAD data is used during training, but testing data has no SQuAD examples

In Table 4, we can see that data from q_{known} does little to improve the QA performance on q_{unk} samples, with the only exception being SearchQA, where data from all the 5 OOD datasets boosts the performance of the model by at least 29%. This strengthens our motivation to use calibrator: although a limited number of OOD examples may not help increase the accuracy of QA models, it can help model to be more certain about the correctness of its predictions, preventing it from answering questions it might not be able to answer.

Finally, if we take the average across 20 settings (excluding the diagonal entries where training and testing use the same OOD dataset), we can obtain a performance comparison across all 3 methods, as in Table 5, which highlights that calibrator has a much superior performance in both of our evaluation metrics.

[2] It is worth noting that the in-domain calibration curve consistently suffers from under-confidence problem, which is cause by the settings of SQuAD: only one answer is provided in training time but multiple valid answers can be accepted during the test time.

	AURC	Cov@80	Cov@90
Original MaxProb	20.25	50.24	22.14
Retuned MaxProb with OOD	19.85	51.65	23.13
Calibrator	18.23	57.73	30.83

Table 5: Average of experiment results across 20 settings

5 Analysis

In this section, we investigate how the calibrator improves its performance on confidence modeling.

5.1 Ablation Study

Firstly, we try to remove features from our calibrator to see how the performance will change, other settings remain the same: Table 6 shows that softmax probabilities still play a crucial role in confidence estimation. Attention features greatly help calibration task as well. The lengths of question and answer have limited effect on the results.

	AURC	Cov@80	Cov@90
All features	18.23	57.73	30.83
-Question Len	18.35	57.32	30.64
-Answer Len	18.24	57.68	30.83
-Top 5 Softmax	18.63	56.17	28.74
-Softmax Variance	18.25	57.65	30.88
-Attention Scores	19.68	53.80	24.84
-Q&A Len	18.35	57.27	30.60
-Softmax&Variance	24.46	34.43	6.8

Table 6: Averaged results with features removed

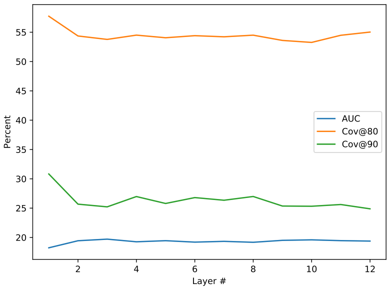


Figure 3: Performance using attention of different layer

5.2 Attention Scores

Our calibrator in the main results uses 12 attention features from the 12 heads of BERT's first layer. We tried across attention features of different layers to see how the calibration performance change. From Figure 3 we can see that the first layer have dominant performance over other layers. Features from other layers have roughly the same performance on the three chosen metrics, which indicates that important information about confidence modeling may be lost after the first layer.

5.3 Discussion about the Domain Shift Setting

At this point our experiments use data mixed with in-domain and out-of-domain examples during the training and testing phase. To have a hint of what contributes to the improvement of calibration task, we report the results on ID and OOD data separately:

Test on ↓	AURC	Cov@80	Cov@90
$p_{source} + q_{unk}$	20.25	50.24	22.14
Only p_{source}	6.58	100	74.05
Only q_{unk}	44.13	9.99	2.11

(a) MaxProb

AURC	Cov@80	Cov@90
18.23	57.73	30.83
6.41	100	77.33
44.18	7.23	2.25

(b) Calibrator

From the table above, we can see that calibrator cannot help the calibration task on p_{source} and q_{unk} only, but have significant improvements when tested on the combined dataset. Considering that our metrics measure the *relative order* of confidence score, this implies that our model is good at adjusting the relative order of confidence scores between p_{source} and q_{unk} examples, but not very helpful in adjusting the relative order within each domain.

In our experiment settings, the ratio of ID and OOD examples is 1, which may not be the case in many real-world scenarios. But our findings show that the calibrator will have better performance when ID example is no less frequent than the OOD one, since our model has better performance when test data has only ID examples or is a equal mix of the two.

We also investigate situations where we only use ID or OOD data to train the calibrator. Experiment results demonstrates that calibrator in these settings has worse performance compared to calibrator trained with both ID and OOD data. This supports our argument that the calibrator is most useful in adjusting the relative order between OOD and ID confidence scores, which is enabled by the mixing of training dataset.

Test on $\downarrow$	AURC	Cov@80	Cov@90
$p_{source} + q_{unk}$	20.08	51.32	22.55
Only p_{source}	6.42	100	76.38
Only q_{unk}	44.24	10.25	1.24

(a) Calibrator trained with only p_{source}

Test on $\downarrow$	AURC	Cov@80	Cov@90
$p_{source} + q_{unk}$	20.84	48.68	20.08
Only p_{source}	7.18	100	74.08
Only q_{unk}	44.94	5.92	0.89

(b) with only q_{known}

5.4 OOD Detection for Calibration

With the finding in previous section, it is natural to connect the task of calibration under domain shift with the task of OOD detection, where a model is asked to identify whether an example is from a different distribution of the training set. Here we conducted two simple experiments:

1. The confidence score of the exmaple is 0 if it is OOD, otherwise we use the MaxProb value. That is, we reject to answer any question from OOD.
2. The calibrator has an extra domain feature indicating whether the example is from ID or OOD.

Model$\downarrow$	AURC	Cov@80	Cov@90
Calibrator	18.23	57.73	30.83
OOD Reject	17.41	58.31	37.90
Calibrator with f_{dom}	16.71	60.7	39.07

Table 7: Calibrator with domain feature

We can see from Table 7 that simply rejecting OOD examples will significantly increase the calibration performance. Furthermore, domain feature has significant improvements on calibrator. Although in real world settings, models are unable to know domain label, the experiment shed light on future works where we can leverage OOD detectors predictions of domain label to boost the calibrator's performance.

6 Conclusion

In this paper, we introduced the problem of confidence modeling under domain shift and conducted various studies to tackle this problem. We reach the conclusion that a calibrator trained on a mixture of in-domain and known OOD data are able improve the system's confidence modeling capability on almost any other unknown OOD data. This observation is significant in that a more accurate confidence modeling enables the system to correctly abstain under low confidence given arbitrary user input, and also guarantee a high accuracy on the question that the system choose to answer.

For future work, we plan to explore additional representative features from the QA model that can enhance the calibrator's understanding of the QA model's confidence to its output, such as the hidden state of the BERT model, a PCA analysis on the self-attention components, or feature combination with OOD detectors. We also hope to extend this calibration technique to other NLP tasks and evaluate its effectiveness as well.

Acknowledgements

Our project is inspired by the work [30] of Amita Kamath and Robin Jia. We would like to extend our gratitude and appreciation to our mentor Robin and Amita for their great help along our journey of writing this report. This project would not be possible without their insightful instruction to our idea and directions.

References

[1] Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomas Kocisky, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, et al. Learning and evaluating general linguistic intelligence. *arXiv preprint arXiv:1901.11373*, 2019.

[2] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*, 2017.

[3] Timothy J Hazen, Shehzaad Dhuliawala, and Daniel Boies. Towards domain adaptation from limited data for question answering using deep neural networks. *arXiv preprint arXiv:1911.02655*, 2019.

[4] Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang, and Mo Yu. Out-of-domain detection for low-resource text classification tasks. *arXiv preprint arXiv:1909.05357*, 2019.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[6] Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of 2nd Machine Reading for Reading Comprehension (MRQA) Workshop at EMNLP*, 2019.

[7] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622, 2015.

[8] Y Gal and Z Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *33rd International Conference on Machine Learning, ICML 2016*, volume 3, pages 1651–1660, 2016.

[9] Yonatan Geifman, Guy Uziel, and Ran El-Yaniv. Bias-reduced uncertainty estimation for deep neural classifiers. In *International Conference on Learning Representations*, 2019.

[10] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016.

[11] Amit Mandelbaum and Daphna Weinshall. Distance-based confidence score for neural network classifiers. *arXiv preprint arXiv:1709.09844*, 2017.

[12] Li Dong, Chris Quirk, and Mirella Lapata. Confidence modeling for neural semantic parsing. In *ACL*, 2018.

[13] Shuo Wang, Yang Liu, Chao Wang, Huanbo Luan, and Maosong Sun. Improving back-translation with uncertainty-based confidence estimation. *arXiv preprint arXiv:1909.00157*, 2019.

[14] Lixin Su, Jiafeng Guo, Yixin Fan, Yanyan Lan, and Xueqi Cheng. Controlling risk of web question answering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–124, 2019.

[15] Ran El-Yaniv and Yair Wiener. On the foundations of noise-free selective classification. *J. Mach. Learn. Res.*, 11:1605–1641, 2010.

[16] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in neural information processing systems*, pages 4878–4887, 2017.

[17] Hady Elsahar and Matthias Gallé. To annotate or not? predicting performance drop under domain shift. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2163–2173, Hong Kong, China, November 2019. Association for Computational Linguistics.

[18] Jean Feng, Arjun Sondhi, Jessica Perry, and Noah Simon. Selective prediction-set models with coverage guarantees. *arXiv preprint arXiv:1906.05473*, 2019.

[19] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6402–6413. Curran Associates, Inc., 2017.

[20] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[22] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[23] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[24] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset. *CoRR*, abs/1611.09830, 2016.

[25] Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *CoRR*, abs/1705.03551, 2017.

[26] Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. Searchqa: A new q&a dataset augmented with context from a search engine. *CoRR*, abs/1704.05179, 2017.

[27] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

[28] Ran El-Yaniv and Yair Wiener. On the foundations of noise-free selective classification. *J. Mach. Learn. Res.*, 11:1605–1641, August 2010.

[29] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

[30] Amita Kamath, Robin Jia, and Percy Liang. Selective question answering under domain shift. In *Association for Computational Linguistics (ACL)*, 2020.