Predicting generalization performance with correctness discriminators

Yuekun Yao and **Alexander Koller**
Department of Language Science and Technology
Saarland Informatics Campus
Saarland University, Saarbrücken, Germany
{ykyao, koller}@coli.uni-saarland.de

Abstract

The ability to predict an NLP model's accuracy on unseen, potentially out-of-distribution data is a prerequisite for trustworthiness. We present a novel model that establishes upper and lower bounds on the accuracy, without requiring gold labels for the unseen data. We achieve this by training a *discriminator* which predicts whether the output of a given sequence-to-sequence model is correct or not. We show across a variety of tagging, parsing, and semantic parsing tasks that the gold accuracy is reliably between the predicted upper and lower bounds, and that these bounds are remarkably close together.

1 Introduction

A prerequisite for the trustworthiness of NLP systems is that a user needs to be able to judge their accuracy on real-world tasks of interest. While neural models have greatly improved the accuracy of NLP systems on benchmarks involving in-distribution test sets, their accuracy on out-of-distribution (OOD) test sets and unseen domains lags behind (Lake and Baroni, 2018; Li et al., 2023). It seems realistic that a user who wants to estimate whether the system's accuracy is sufficient for their purposes could produce natural-language input that reflects their particular use case; it is less realistic that an untrained user would annotate these inputs with gold outputs that would allow them to directly establish the system's accuracy.

There is some previous work that estimates a model's accuracy on unseen OOD data without requiring gold annotations, mostly for image and text classification tasks (Garg et al., 2022; Guillory et al., 2021). Existing methods provide a point estimate for the model's accuracy, often by exploiting statistical properties of the model's confidence on the unseen inputs. However, this still leaves the user uncertain about the accuracy of the point estimate: Can we rely on the accuracy of the accuracy

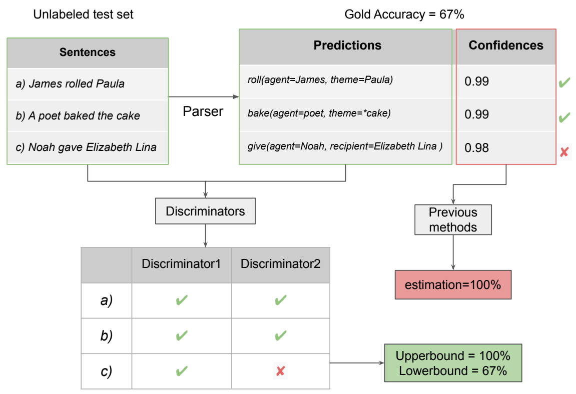


Figure 1: Comparison of our discriminators and confidence-based methods. Our method provides upper and lower bounds which can capture gold accuracy.

estimate, or should we assume a wider potential margin for error?

In this paper, we present a method for predicting *upper and lower bounds* for the accuracy of a model on unlabeled test data. We focus on sequence-to-sequence models, applied to parsing, semantic parsing, and tagging tasks; these tasks have the advantage over other sequence generation tasks that there is a unique correct answer, which allows us to talk about accuracies. We first train a *discriminator* to predict whether the model's output on a given input is correct or not; we show that this can be done with remarkable accuracy across a range of tasks.

We then run an ensemble of discriminators on the predictions of fine-tuned T5 models (Raffel et al., 2020) on the unlabeled test data and obtain upper and lower bounds through a voting mechanism (Figure 1). We show across a variety of in-distribution and OOD tasks that the model's true accuracy is reliably between the upper and lower bounds, and that these bounds are quite tight. Finally, we show that across most datasets, the mean of the upper and lower bounds provides a more precise point estimate of the true accuracy than earlier work.

2 Related work

Calibration. Calibration is one of the most related directions to our work. Generally, a neural model is *well-calibrated* if its predicted probability (e.g. confidence) for its decision (e.g. label or sequence) aligns to the probability of the prediction correctness. Such calibration works rely on either modifying training objectives or posthoc methods: Kong et al. (2020) add a regularization term into training objective to address in-distribution calibration and out-of-distribution detection for text classification; Desai and Durrett (2020) exploit temperature scaling (Guo et al., 2017) to normalize output logits with a scalar temperature parameter; Dong et al. (2018); Kamath et al. (2020) train an additional regressor to estimate the model confidence with designed features for semantic parsing; Jiang et al. (2021b) investigate all these methods and find that posthoc-based methods are universally helpful for question answering tasks.

Most calibration works above focus on in-distribution (ID) tasks and assume a development set is given, which allows them to estimate parameters (e.g. temperature) to yield the optimal confidence. However, according to Kamath et al. (2020), the predicted model confidence is an unreliable estimate of the correctness on OOD generalization tasks. Compared to such calibration works, our method applies just as easily to OOD as to ID tasks. Further, development sets from OOD distributions are usually difficult to access, which introduces the additional challenge of applying calibration-based methods. Kamath et al. (2020) also consider distribution shift, but their calibrator requires access to a small amount of data from a known OOD distribution.

Predicting test accuracy from unlabeled data. Previous works have also investigated predicting the performance on an OOD test set for other tasks: Guillory et al. (2021) exploit the difference of confidences between training distribution and the OOD distribution as a useful feature; Jiang et al. (2021a) show that the test error of deep networks can be estimated by the disagreement of two models trained with the same architecture on the same training set but with two different runs; Yu et al. (2022) exploits the euclidean distance between model parameters trained on differently distributed data to predict generalization errors; Garg et al. (2022) estimate a threshold of model confidence from training data and predict the correctness of OOD data based on it; Fu et al. (2023) train an additional model to predict the accuracy of large language models on question answering tasks, which takes as input confidence scores and outputs the overall accuracy of the test set.

Most works mentioned above predict model accuracy on OOD data based on unlabeled data from OOD distributions. However, these works only consider image classification and natural language inference tasks. In our work, we show that for sequence generation tasks like semantic parsing, the predicted sequence can serve as a good-enough feature to determine the prediction correctness on OOD data. Besides, our method naturally yields an upper and lower bounds for the predicted accuracy, in contrast to previous methods which only provide a point estimation.

3 Correctness discriminator

The core of our methodology is the construction and training of a correctness discriminator model, which judges the correctness of a model prediction on unseen data. In this section we first introduce how we design the discriminator model and collect training data in Section 3.1, and then describe how to predict the upper bound and lower bounds accuracy in Section 3.2. To avoid confusion, we call the model for the original parsing or tagging tasks a *parser* and the model for predicting the parser performance a *discriminator*. Note that here we only assume that the parser solves a sequence-to-sequence task, but the task output can be any sequence – not just a linearized parse tree.

3.1 Discriminator design

The discriminator is designed as a binary classifier whose task is to determine whether a given pair of a natural language sentence and a corresponding predicted sequence is correctly matched. Formally, given a natural language sentence $X \in \mathcal{X}$ and a predicted symbolic sequence (e.g. meaning representation for semantic parsing tasks) $Y \in \mathcal{Y}$, the discriminator $F : \mathcal{X} \times \mathcal{Y} \rightarrow \{Correct, Incorrect\}$ maps them to a *Correct* or *Incorrect* label to represent its correctness.

In this paper, we use an encoder-decoder pretrained language model (e.g. T5, BART) as our discriminator, where the encoder takes as input the concatenation of the input natural language sen-

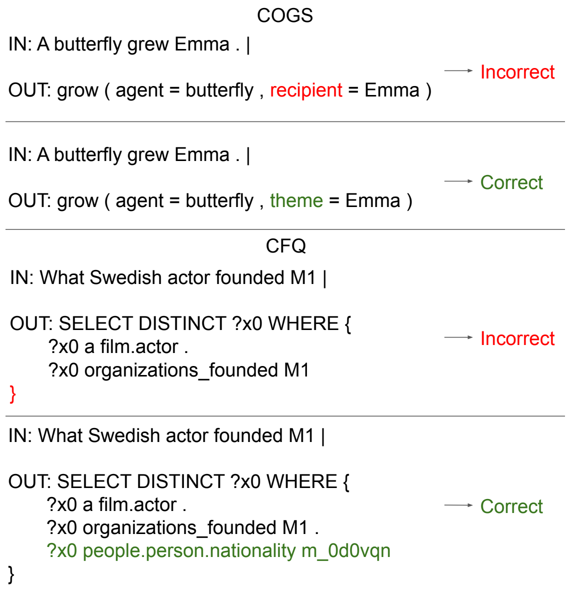


Figure 2: Examples of COGS and CFQ training data for the discriminator. *IN* refers to the input sentence and *OUT* refers to the predicted output sequence (e.g. logical form for COGS and SPARQL query for CFQ).

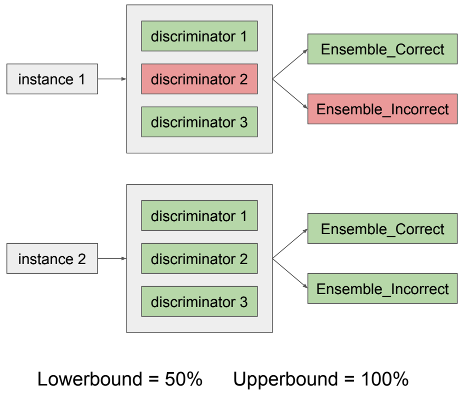


Figure 3: Example to illustrate how we calculate upper and lower bounds from discriminators. *Green* blocks mean the instance is predicted as *Correct* by the discriminator and *Red* blocks refer to *Incorrect*. For instance 1, the discriminator 1 predicts *Correct* and discriminator 2 predicts *Incorrect*. Thus, the ensemble prediction is *Correct* using *ensemble_correct* voting mechanism and *Incorrect* using *ensemble_incorrect*.

tence with the predicted sequence, and the decoder generates *Correct* or *Incorrect* label.

Now we introduce how to collect training data for our discriminator. In principle, the training data should contain both positive and negative examples. For positive examples, we can always exploit the training set used to train the parser. However, it is non-trivial to obtain negative examples. Such examples can be synthesized by applying noise functions (e.g. replacement or deletion) to positive examples (Kim et al., 2021), but this requires prior knowledge about errors a parser tends to make. Another option is to collect errors a trained parser made on its training set, which is still challenging since the parser easily yields perfect accuracy on its training set in our pilot study.

Due to reasons above, we decide to generate negative examples with intermediate checkpoints of our parser during its training. Specifically, we validate the parser checkpoint on its training set before the parser achieves perfect accuracy. Our parser is implemented with an encoder-decoder model, and thus we can collect incorrect predictions from outputs of the beam search as negative examples. Figure 2 gives examples of our training data.

3.2 Bounds prediction

With the correctness discriminator described above, we can now how to predict the upper and lower bounds accuracy of our parser on the test set. This is implemented by ensembling multiple discriminators with two novel voting mechanisms. Specifically, we ensemble discriminator predictions such that an instance is labeled as *Incorrect* if there exists a discriminator prediction of this instance being *Incorrect*. We call this vote mechanism *ensemble_incorrect*, which yields a lowerbound for the predicted accuracy, since it aims to capture all possible incorrect predictions. Similarly, we can define *ensemble_correct* by labeling an instance *Correct* if there exists a discriminator prediction being *Correct*, which yields an upper bound of the predicted accuracy. Figure 3 shows an example of how to estimate such bounds. The ensemble of deep neural networks have been shown to be effective for uncertainty quantification (Lakshminarayanan et al., 2017; Lukovnikov et al., 2021). Different from previous works where model predictions are combined by averaging predicted probabilities, we use two hard voting mechanisms to calculate the upper and lower bounds of our predicted accuracy.

4 Experiments

We introduce our datasets, model setup, evaluation metrics and experimental results in this section.

4.1 Datasets

We mainly consider two OOD generalization problems in our experiments: compositional generalization (Lake and Baroni, 2018) with COGS, POS-

COGS and CFQ, and low-resource domain adaptation (Chen et al., 2020) with TOPv2. Additionally, we apply our method to in-distribution tasks, including AMR and Penn Treebank 3, to see its effect.

COGS (Kim and Linzen, 2020) is a synthetic English semantic parsing task. The task input is a sentence and the output is a logical form (e.g. *The baby on a tray in the house screamed.* $\rightarrow$ `scream(agent=*baby(nmod.on=tray(nmod.in=*house)))`). It provides a training set generated by a probabilistic context-free grammar (PCFG) and a OOD test set with 21-typed data, which are generated by different PCFGs to test the different generalization abilities of the parser.

POS-COGS (Yao and Koller, 2022) is a synthetic English part-of-speech tagging task generated based on COGS. The task input is a sentence and the output is the POS tag sequence (e.g. *The baby on a tray in the house screamed.* $\rightarrow$ `Det N P Det N P Det N V`). POS-COGS shares the same split of train and test sets as COGS.

CFQ (Keysers et al., 2020) is a synthetic English semantic parsing task. The task input is a sentence and the output is a SPARQL query (e.g. *Did M0 ' s writer write M1 and M2* $\rightarrow$ `SELECT count(*) WHERE {?x0 film.writer.film M0...}`). We use the MCD1 and MCD2 splits of CFQ, where the test set is designed to compositionally diverge from the training set but still share similar atom distributions.

TOPv2 (Chen et al., 2020) is a natural English semantic parsing task. The task input is a sentence and the output is a hierarchical semantic representation (Gupta et al., 2018) (e.g. *Will there be snowfall this week?* $\rightarrow$ `[in:get_weather will there be [sl:weather_attribute snowfall] [sl:date_time this week] ?]`). The TOPv2 training set consists of data from multiple domains including two low-resource domains (e.g. *reminder* and *weather*), and the test set consists of data from the two domains to test low-resource domain adaptation ability of the parser. We focused on the*weather* domain in our experiments.

AMR (Banarescu et al., 2013) is a natural English semantic parsing task. The task input is a sentence and the output is an abstract meaning representation (e.g. *I will stick around until the end.* $\rightarrow$ `(stick-around-03 :ARG0(i) :time(until :op1(end-01)))`). We use AMR 2.0 in experiments.

Penn Treebank 3 (PTB) (Marcus et al., 1993) is an English constituency parsing task. The task input is a sentence and the output is the constituency parse tree (e.g. *Vice President* $\rightarrow$ `(TOP(NP(NNP Vice)(NNP President)))`). We train our parser on the WSJ training set and evaluate its in-domain performance on the WSJ test set and cross-domain performance on the Brown corpus (Marcus et al., 1993). We predict the generalization performance for both WSJ test set and Brown corpus, which we call Syn-WSJ, Syn-Brown (for parsing) and POS-WSJ, POS-Brown (for tagging) in the rest of the paper.

4.2 Setup

Parser. We finetune *T5-base* as the parser for all tasks described above. To do this, we convert all of our tasks into sequence generation tasks, where the output sequence can be a semantic meaning representation, POS tag sequence or a linearized constituency parse tree. All our parsers achieve the same or close performance as those reported in previous works using T5. For CFQ and PTB, there exist some special tokens which are not in the vocabulary of T5 so we replaced them with in-vocabulary tokens.

Discriminator. We also finetune another *T5-base* as the discriminator for each task. To collect positive examples of training data for the discriminator, we concatenate the input sentence and gold output sequence for each instance in the parser training set. For negative examples, we validate the parser checkpoint every K steps on its training set, where K is a hyperparameter. Since our parser is an encoder-decoder model, we randomly sample incorrect predictions from the decoded beam predictions. In our experiments, we use the first 5 checkpoints and combine data generated from them as the training set. If a task provides an in-distribution development set for the parser, we use the same method to create the development set for the discriminator. Otherwise, we train the discriminator until its training loss converges with fixed steps.

Comparable baseline. We also compare our methods with several previous methods.

MaxProb. Maxprob is a strong baseline shown in Kamath et al. (2020). Assuming we are given a threshold γ on the maximal prediction probability (e.g. Confidence) of a parser, we can predict an instance as *Correct* if the parser confidence on this instance is higher than γ, otherwise *Incorrect*. Since we have no prior knowledge about the OOD

		Single			Upper bound			Lower bound			Gold
		CR	IR	Acc	CR	IR	Acc	CR	IR	Acc	Acc
OOD	MCD1	97.0	83.1	63.1	**99.2**	69.8	70.0	92.9	**94.5**	56.0	57.8
	MCD2	80.6	83.7	31.0	**83.4**	77.9	36.1	71.7	**92.2**	22.4	22.9
	COGS	98.5	96.6	90.3	**99.8**	89.4	92.1	98.5	**96.9**	90.2	91.4
	TOP	87.4	57.9	75.2	**92.2**	44.7	82.3	82.5	**78.9**	66.0	73.0
	POS-Brown	80.2	53.6	67.0	**93.0**	26.5	86.2	52.4	**84.8**	39.5	61.0
	POS-COGS	98.8	86.3	86.6	**99.9**	84.4	87.9	98.7	**89.2**	86.2	85.7
	Syn-Brown	35.3	60.5	38.1	**58.9**	43.2	57.5	11.7	**78.9**	17.9	33.8
ID	AMR	37.0	98.3	6.0	**56.5**	88.1	18.3	21.8	**99.7**	3.4	14.3
	POS-WSJ	81.3	54.7	68.8	**94.0**	26.0	86.2	52.2	**84.7**	37.8	65.3
	Syn-WSJ	43.7	54.2	45.0	**65.9**	36.8	64.2	20.4	**72.8**	24.6	37.6

Table 1: Results of our discriminators on different datasets. For each dataset, we report *Correct-Recall* (CR), *Incorrect-Recall* (IR), and predicted accuracy (Acc). *Single* refers to the results with predictions from a single discriminator. *Upperbound* refers to the results with discriminator predictions using *ensemble_incorrect* and the similar for *Lowerbound*. *Gold* refers to the accuracy evaluated with gold annotations.

distribution, we set $\gamma = 0.5$ in our experiments.

Average Confidence (AC). We take the average confidence across the test set as the predicted accuracy. Different from previous works where the confidence is defined as the maximal softmax probability of the classifier, here we define the confidence as the probability of the most possible sequence in the beam, which is calculated by the product of softmax probabilities of each word in the sequence.

Average Thresholded Confidence (ATC) is a strong method recently proposed by Garg et al. (2022), which has been shown to be more effective than previous methods. Applying ATC consists of two steps. First, we estimate a threshold γ on parser confidence scores to make the number of errors made by the parser match the number of instances where the parser confidence is lower than γ; then we can obtain the predicted accuracy on the test set by calculating the fraction of unlabeled instances that obtain a score below γ.

Maxprob (Oracle). To compare with the upper and lower bounds of our discriminators, we also calculate bounds based on Maxprob, where we assume we can access gold annotations of the test set to estimate γ such that the *Correct-Recall* calculated based on γ is equal to the one from the predicted upper bound calculated by our discriminators. Similarly, we can calculate a lower bound by matching *Incorrect-Recall* scores of the test set.

4.3 Evaluation metrics

For all parser tasks, we evaluate the exact match accuracy of our parser.

For discriminators, we need a metric to quantify the quality of the predicted upper and lower bounds. Intuitively, such a metric should reflect *whether the gold accuracy is within the bounds* (i.e. reliability) and *whether the bounds are tight* (i.e. tightness).

Different from our work, previous work predicts OOD performance by making a point estimation and then evaluating their method with mean absolute estimation error (MAE) by calculating average absolute difference between the true accuracy on the target data and the estimated accuracy on the same unlabeled examples. Their results are averaged over multiple test sets for each classifier (e.g. parser in our tasks). In our setup, most tasks only have one OOD or ID test set, and thus we directly calculate the absolute estimation error (AE) to compare with previous works. Equation 1 defines the metric, where Acc_{gold} denotes the gold accuracy and Acc_{pred} denotes the predicted accuracy.

$$|Acc_{gold} - Acc_{pred}| \quad (1)$$

We simply calculate the mean of our predicted upper and lower bounds as the point accuracy estimation (Acc_{pred}). Despite its simplicity, we find this method performs well across our tasks.

In addition, we report *Precision* and *Recall* of our the discriminator individually. For *Precision* and *Recall*, we report the score for the *Correct*

	OOD								ID	
	MCD1		MCD2		COGS		TOP		AMR 2.0	
	Acc	AE ↓	Acc	AE ↓	Acc	AE ↓	Acc	AE ↓	Acc	AE ↓
Maxprob	84.5	26.7	78.0	55.1	97.1	5.7	92.2	19.2	40.6	26.3
AC	82.5	24.7	74.0	51.1	96.6	5.2	85.9	12.9	38.0	23.7
ATC	73.0	15.2	56.9	34.0	100.0	8.6	66.0	7.0	15.0	**0.7**
Maxprob (Oracle)										
Upper.	86.4	-	51.3	-	96.7	-	85.8	-	17.9	-
Lower.	43.7	-	17.2	-	44.3	-	65.2	-	5.8	-
Mean	65.1	7.3	34.3	11.4	70.5	20.9	75.5	2.5	11.8	2.5
Ours										
Single	63.1	5.3	31.0	8.1	90.3	1.1	75.2	2.2	6.0	8.3
Upper.	70.0	-	36.1	-	92.1	-	82.3	-	18.3	-
Lower.	56.0	-	22.4	-	90.2	-	66.0	-	3.4	-
Mean	63.0	**5.2**	29.3	**6.4**	91.2	**0.3**	74.2	**1.2**	10.9	3.4
Gold	57.8	0.0	22.9	0.0	91.4	0.0	73.0	0.0	14.3	0.0

Table 2: Predicted test-set accuracy with different methods on semantic parsing tasks. For each dataset, we report predicted accuracy (Acc) and AE scores. *Upper.* and *Lower.* in the leftmost column refer to predicted upper bound and lower bound. *Gold* refers to the accuracy evaluated with gold annotations.

and *Incorrect* labels individually. We define *Truely Correct (TC)* as instances with an annotation being *Correct* and the prediction being *Correct*, *Falsely Correct (FC)* as instances with an annotation being *incorrect* and the prediction being *correct*. Similarly, we can define *Truely Incorrect* (TI) and *Falsely Incorrect* (FI). Thus the *Correct-Recall* is calculated by Equation 2. The scores for *Incorrect* label is calculated the same way. These precision and recall scores indicate how many instances can be correctly or incorrectly discriminated, but this is not studied by previous works. Here we propose these metrics as a side contribution of our work, which can be beneficial for downstream uses of the discriminator. Due to space constraints, we report *Correct-Recall* and *Incorrect-Recall* in the main paper and report the *Precision* results in the appendix.

$$Count(TC)/(Count(TC) + Count(FI)) \quad (2)$$

4.4 Results

We first report results of our discriminators in Table 1. First, we can observe that the predicted accuracy from a single discriminator is already close to the gold accuracy for most datasets. This shows that our trained discriminator does learn to discriminate predictions from errors made on training sets by the parser checkpoints, and this discrimination ability can generalize well to OOD test sets.

Correctness prediction of bounds. According to Table 1, we can also observe that the upper bound achieves the highest *Correct-Recall* score, and lower bound achieves the highest *Incorrect-Recall* score. This is because these bounds are based on voting mechanisms specifically designed to find correct or incorrect predictions. On many of our datasets, these recall scores approach 100%, which indicates the strong ability of our method to discriminate correctness.

Accuracy prediction of bounds. We also compare the predicted accuracy of our bounds in Table 2 (e.g. semantic parsing), Table 3 (e.g. tagging) and Table 4 (e.g. parsing). We can observe that our predicted upper and lower bounds accurately capture the gold accuracy (i.e. high reliability). This pattern holds for 9 of 10 datasets, and even for POS-COGS, where this conclusion is not true, the gold accuracy only violates the bounds by a small amount. Meanwhile, the predicted upper and lower bounds are usually close (e.g. high tightness). Comparing our predicted bounds with *Maxprob (Oracle)*, our bounds are more tight on OOD generalization tasks (e.g. MCD splits and COGS). Note that *Maxprob (Oracle)* can access gold annotations to find a proper bound, which is not possible in the real world. Nonetheless, our method still provides better bounds than this oracle method, indicating the effectiveness of our method on OOD tasks.

We also compare our method with other point estimation methods by using the mean of bounds

	OOD				ID	
	POS-Brown		POS-COGS		POS-WSJ	
	Acc	AE	Acc	AE	Acc	AE
Maxprob	87.4	26.4	99.8	14.1	84.7	19.4
AC	80.5	19.5	100.0	14.3	77.4	12.1
ATC	68.0	7.0	100.0	14.3	61.6	3.7
Maxprob (Oracle)						
Upper.	83.6	-	99.6	-	82.4	-
Lower.	44.5	-	83.3	-	47.9	-
Mean	64.0	3.0	91.4	5.7	65.2	**0.1**
Ours						
Single	67.0	6.0	86.6	**0.9**	68.8	3.5
Upper.	86.2	-	87.9	-	86.2	-
Lower.	37.8	-	86.2	-	39.5	-
Mean	62.0	**1.0**	87.1	1.4	62.9	2.4
Gold	61.0	0.0	85.7	0.0	65.3	0.0

Table 3: Predicted test-set accuracy on POS tagging tasks.

	OOD		ID	
	Syn-Brown		Syn-WSJ	
	Acc	AE	Acc	AE
Maxprob	48.3	14.5	50.8	13.2
AC	50.8	17.0	52.4	14.8
ATC	34.7	**0.9**	34.0	**3.6**
Maxprob (Oracle)				
Upper.	32.9	-	33.4	-
Lower.	17.7	-	16.5	-
Mean	25.3	8.5	24.9	12.7
Ours				
Single	38.1	4.3	45.0	7.4
Upper.	57.5	-	64.2	-
Lower.	17.9	-	24.6	-
Mean	37.7	3.9	44.4	6.8
Gold	33.8	0.0	37.6	0.0

Table 4: Predicted test-set accuracy on constituency parsing tasks.

as our predicted accuracy (e.g. *Mean* row in *Ours*). Although our method is not specifically designed for point estimation, it substantially outperforms previous methods and achieves a relatively low AE score on semantic parsing and POS tagging tasks. On constituency parsing tasks, our method does not outperform *ATC* (Garg et al., 2022), but is still better than other baselines. Our method is also especially useful for OOD test sets, where confidence-based methods yield a much larger AE.

5 Discussion

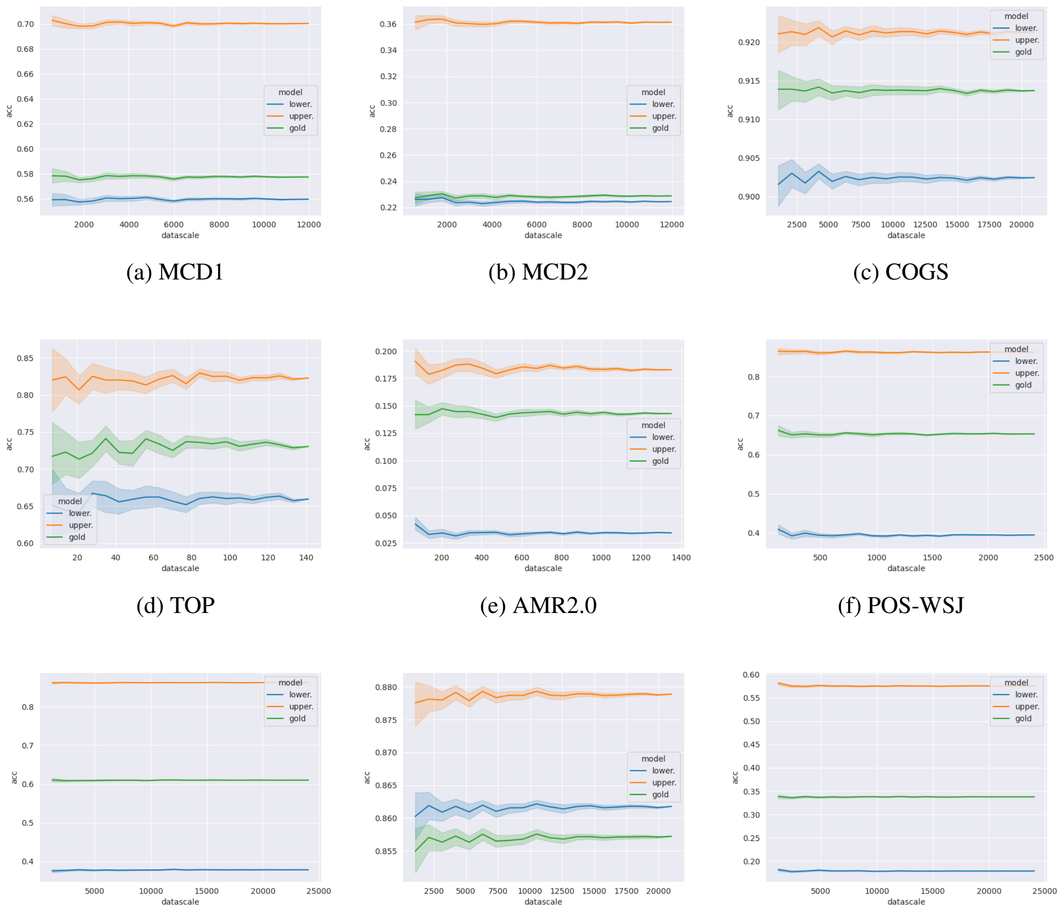
Low performance on constituency parsing. Our method does not outperform *ATC* (Garg et al., 2022) on PTB parsing tasks. We consider this is because the PTB training set contains many long output sequences (e.g. linearized parse trees), whose lengths are much larger than the maximal encoding length (e.g. 512) of our language model discriminators. Encoding sequences longer than observed during pretraining has been shown challenging for such transformer-based language models (Dai et al., 2019), which leads to an additional challenge for our discriminators. Nonetheless, our method still yields upper and lower bounds that can capture the gold accuracy.

The robustness of discriminators We have seen that our predicted upper and lower bounds accuracy can capture the gold accuracy. However, this may not be enough to show the robustness of our method, since we only evaluated it on one overall test set for each parser, while previous works (Garg et al., 2022) collect multiple test sets for each classifier. To investigate the robustness of our predicted bounds, we further create multiple test sets by randomly sampling subsets from the original test set with different and plot the accuracy in Figure 4.

According the results, we can observe that our predicted bounds robustly capture the gold accuracy with regard to different sizes of randomly sampled test sets. On COGS, POS-COGS and TOP, a small test set gives a large confidence interval. We consider this is because their test sets contain some extremely difficult examples for the parser, which could result in a very challenging subset and yield a low accuracy. Despite this, our discriminator can capture the difficulty of such challenging subsets and shares similar confidence intervals as the gold accuracy.

6 Conclusion

We propose a method to predict *upper and lower bounds* for the accuracy of a model on unlabeled out-of-distribution data. To do this, we first train multiple correctness discriminators implemented by a pretrained encoder-decoder language model, and then ensemble discriminator predictions through a special voting mechanism. Our experiments show that our predicted bounds reliably capture gold accuracy across a variety of in-distribution and out-of-distribution tasks including semantic parsing, tagging and constituency parsing



(a) MCD1 (b) MCD2 (c) COGS

(d) TOP (e) AMR2.0 (f) POS-WSJ

(g) POS-Brown (h) POS-COGS (i) Syn-Brown

Figure 4: Predicted accuracy w.r.t. the number of test-set instances. For each subset we randomly sample 50 times and show its confidence interval with 95% confidence.

tasks and the upper and lower bounds are usually close. Although our method is not specifically designed for point estimation, a simple heuristic (e.g. using the mean of bounds as estimated accuracy) based on our method can substantially outperform previous methods, which indicates the effectiveness of our method.

For the future, we will explore the use of our discriminators to improve model performance on tasks evaluated in this paper. For example, given unlimited out-of-distribution natural language sentences and a parser, our lower bound can be used to detect instances with a high *Correct-Precision*, which can be used as training data to further train the parser. It will also be interesting to explore whether our method can be extended to other tasks by predicting different metrics (e.g. BLEU) instead of exact match accuracy.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020. Low-resource domain adaptation for compositional task-oriented semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100, Online. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 295–302, Online. Association for Computational Linguistics.

Li Dong, Chris Quirk, and Mirella Lapata. 2018. Confidence modeling for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 743–753, Melbourne, Australia. Association for Computational Linguistics.

Harvey Yiyun Fu, Qinyuan Ye, Albert Xu, Xiang Ren, and Robin Jia. 2023. Estimating large language model capabilities without labeled test data.

Saurabh Garg, Sivaraman Balakrishnan, Zachary C Lipton, Behnam Neyshabur, and Hanie Sedghi. 2022. Leveraging unlabeled data to predict out-of-distribution performance. *arXiv preprint arXiv:2201.04234*.

Devin Guillory, Vaishaal Shankar, Sayna Ebrahimi, Trevor Darrell, and Ludwig Schmidt. 2021. Predicting with confidence on unseen distributions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1134–1144.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792, Brussels, Belgium. Association for Computational Linguistics.

Yiding Jiang, Vaishnavh Nagarajan, Christina Baek, and J Zico Kolter. 2021a. Assessing generalization of sgd via disagreement. *arXiv preprint arXiv:2106.13799*.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. 2021b. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Amita Kamath, Robin Jia, and Percy Liang. 2020. Selective question answering under domain shift. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5684–5696, Online. Association for Computational Linguistics.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations (ICLR)*.

Juyong Kim, Pradeep Ravikumar, Joshua Ainslie, and Santiago Ontanon. 2021. Improving compositional generalization in classification tasks via structure annotations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 637–645, Online. Association for Computational Linguistics.

Najoung Kim and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105, Online. Association for Computational Linguistics.

Lingkai Kong, Haoming Jiang, Yuchen Zhuang, Jie Lyu, Tuo Zhao, and Chao Zhang. 2020. Calibrated language model fine-tuning for in- and out-of-distribution data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1326–1340, Online. Association for Computational Linguistics.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2873–2882, Stockholmsmässan, Stockholm Sweden. PMLR.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30.

Bingzhi Li, Lucia Donatelli, Alexander Koller, Tal Linzen, Yuekun Yao, and Najoung Kim. 2023. Slog: A structural generalization benchmark for semantic parsing.

Denis Lukovnikov, Sina Daubener, and Asja Fischer. 2021. Detecting compositionally out-of-distribution examples in semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 591–598, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Yuekun Yao and Alexander Koller. 2022. Structural generalization is hard for sequence-to-sequence models.

In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5048–5062, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yaodong Yu, Zitong Yang, Alexander Wei, Yi Ma, and Jacob Steinhardt. 2022. Predicting out-of-distribution error with the projection norm. In *International Conference on Machine Learning*, pages 25721–25746. PMLR.