GAPX: Generalized Autoregressive Paraphrase-Identification X

Yifei Zhou
Cornell University
yz639@cornell.edu

Renyu Li
Cornell University
rl626@cornell.edu

Hayden Housen
Cornell University
hth33@cornell.edu

Ser-nam Lim
Meta AI
sernamlim@fb.com

Abstract

Paraphrase Identification is a fundamental task in Natural Language Processing. While much progress has been made in the field, the performance of many state-of-the-art models often suffer from distribution shift during inference time. We verify that a major source of this performance drop comes from biases introduced by negative examples. To overcome these biases, we propose in this paper to train two separate models, one that only utilizes the positive pairs and the other the negative pairs. This enables us the option of deciding how much to utilize the negative model, for which we introduce a perplexity based out-of-distribution metric that we show can effectively and automatically determine how much weight it should be given during inference. We support our findings with strong empirical results. [1]

1 Introduction

Paraphrases are sentences or phrases that convey the same meaning using different wording, and is fundamental to the understanding of languages [7]. Paraphrase Identification is a well-studied task of identifying if a given pair of sentences has the same meaning [51, 47, 56, 57, 31], and has many important downstream applications such as machine translation [61, 44, 40, 27], and question-answering [11, 35].

QQP	PAWS	WMT	PIT
similar topic	same bag-of-word	poor machine translation	similar topics
What causes stool color to change to yellow?	Emma Townshend is represented by David Godwin at DGA Associates .	The company then changed its company ID number, and that was it.	That amber alert was getting annoying
What can cause stool to come out as little balls?	David David Godwin is represented at DGA Associates by Emma Townshend .	But it changed the name and IDN and there was nothing to deal with.	Why do I get amber alerts tho

Figure 1: Negative pairs in different datasets are mined differently in different datasets, and can lead to significant biases during training.

Recently, researchers have observed that neural network architectures trained on different datasets could achieve state-of-the-art performances for the task of paraphrase identification [52, 16, 50]. While these advances are encouraging for the research community, it has however been observed

[1] Our code is publicly available at: https://github.com/YifeiZhou02/generalized_paraphrase_identification

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

that these models can be especially fragile in the face of distribution shift [45]. In other words, when a model trained on a source dataset $\mathcal{D}^s$ is tested on another dataset, $\mathcal{D}^t$, collected and annotated independently, and with a distribution shift, the classification accuracy drops significantly [62].

This paper presents our findings and observations that negative pairs (i.e., non-paraphrase pairs) in the training set, as opposed to the positive pairs, do not generalize well to out-of-distribution test pairs. Intuitively, negative pairs only represent a limited perspective of how the meanings of sentences can be different (and indeed it is practically infeasible to represent every possible perspective). We conjecture that negative pairs are so specific to the dataset that they adversely encourage the model to learn biased representations. We show this observation in Figure 1. Quora Question Pair (QQP) [2] extracts its negative pairs from similar topics. Paraphrase Adversarials from Word Scrambling (PAWS) Zhang et al. [62] generate negative pairs primarily from word swapping. World Machine Translation Metrics Task 2017 (WMT) [8] considers negative examples as poor machine translations. We therefore hypothesize that biases introduced by the different ways negative pairs are mined are major causes of the poor generalizability of paraphrase identification models.

Based on this observation, we would like to be able to control the reliance on negative pairs for out-of-distribution prediction. In order to achieve this, we propose to explicitly train two separate models for the positive and negative pairs (we will refer to them as the positive and negative models respectively), which will give us the option to choose when to use the negative model. It is well known that just training on positive pairs alone can lead to a degenerate solution [21, 59, 42, 13], e.g., a constant function network would still produce a perfect training loss. To prevent this, we propose a novel generative framework where we use an autoregressive transformer [39, 49], specifically BART [30]. Given two sentences, we condition the prediction of the next token in the second sentence on the first sentence and the previous tokens. In a Bayesian sense, this would mean that the next token predicted has a higher probability of being a positive/negative pair to the first sentence for the positive and negative model respectively. This learning strategy has no degenerate solutions even when we are training the positive and negative models separately. We call our proposed approach GAP to stand for Generalized Autoregressive Paraphrase-Identification. One potential pitfall of GAP is that it ignores the "interplay" between positive and negative pairs that would otherwise be learned if they are utilized in training together. This is especially important when the test pairs are in-distribution. To overcome, we utilize an extra discriminative model, trained with both positive and negative pairs, to capture the interplay. We call this extension GAPX (pronounced as "Gaps") to capture the e*X*tra discriminative model used.

For all practical purposes, the weights we placed on the positive, negative and/or discriminative model in GAP and GAPX need to be determined automatically during inference. For in-distribution pairs, we desire to use them all, while for out-of-distribution pairs, we hope to rely on the positive model much more heavily. This obviously leads to a question of how to determine whether a given pair is in or out of distribution [9, 14, 43, 17, 20]. During testing, our method ensembles the positive model, the negative model, and the discriminative model based on the degree of the similarity of the test pair to the training pairs, and found that this works well for our purpose. We measure this degree of similarity with probability cumulative density function (cdf) in terms of perplexity [25], and show that it is superior to other measures.

In summary, our contributions are as follow:

1. We report new research insights, supported by empirical results, that the negative pairs of a dataset could potentially introduce biases that will prevent a paraphrase identification model from generalizing to out-of-distribution pairs.
2. To overcome, we propose a novel autoregressive modeling approach to train both a positive and a negative model, and ensemble them automatically during inference. Further, we observe that the interplay between positive and negative pairs are important for in-distribution inference, for which we add a discriminative model. We then introduce a new perplexity based approach to determine whether a given pair is out-of-distribution to achieve auto ensembling.
3. We support our proposal with strong empirical results. Compared to state-of-the-art transformers in out-of-distribution performance, our model achieves an average of 11.5% and 8.4% improvement in terms of macro F1 and accuracy respectively over 7 different out-

[2] https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

of-distribution scenarios. Our method is especially robust to paraphrase adversarials like PAWS, while keeping comparable performance for in-distribution prediction.

2 Related Works

2.1 Distribution Shift and Debiasing Models in NLP

The issue of dataset bias and model debiasing has been widely studied in a lot of field in NLP such as Natural Language Inference [23, 3] and Question Answering [36, 2, 4]. Notable work by [22, 5, 10, 15, 48] utilize ensembling to reduce models' reliance on dataset bias. These models share the same paradigm where they break down a given sample x into signal feature x_s and biased feature x_b, in the hope of preventing their model from relying on x_b, which has been shown to be the limiting factor preventing the model from generalizing to out-of-distribution samples [5]. Here, a separate model is first either trained on x_b or on datasets with known biases [10, 15, 22], or acquired from models known to have limited generalization capability. Then they train their main model with a regularization term that encourages the main model to produce predictions that deviate from that of the "biased model". However, this type of approach has shown limited success [45] in debiasing paraphrase identification models. In contrast, our method is based on our observation that negative pairs limit the generalization of paraphrase identification models.

2.2 Out-of-distribution Detection

Another line of work relevant to this paper is the task of detecting out-of-distribution samples [9, 14, 43, 17, 20, 55]. Researchers have proposed methods to detect anomaly samples by examining the softmax scores [32] or energy scores [33, 60] produced by discriminative models, while others take a more probabilistic approach to estimate the probability density [14, 25, 66, 1] or reconstruction error [38]. In this paper, we introduce a novel perplexity based out-of-distribution detection method that we show empirically to work well for our purpose. Specifically, during inference, an out-of-distribution score is utilized to weigh the contributions from the positive and negative models: the higher the confidence that the sample is out-of-distribution, the lesser the negative model's contribution.

2.3 Text Generation Metrics

Finally, we would like to note the difference between our work and autoregressive methods that have been explored [58, 46] for evaluating text generation. Our work differs as follows: 1) Paraphrase identification seeks to assign a label of paraphrase or not while text generation metrics seeks to assign a score to measure the similarity of sentences; 2) Current text generation metrics either cannot be trained to fit to a specifc distribution [61, 58, 63] or are limited to the i.i.d. setting [44, 41] of the training distribution. In contrast, our method not only significantly improves out-of-distribution performances but is also competitive with state-of-the-art paraphrase identification methods for in-distribution predictions.

3 Methodology

We observe that negative pairs in paraphrase identification constitute the main source of bias. To overcome this, we propose the following training paradigms to learn a significantly less biased paraphrase identification model. We employ an autoregressive conditional sentence generators with transformer architecture as the backbone of our model. Specifically, we train a positive and negative model to estimate the distribution of positive and negative pairs in a dataset respectively. During testing, the two models are ensembled based on how likely the input pair is out of distribution. This section provides details on our method.

3.1 Separation of Dependence on Positive and Negative Pairs

Let $\mathcal{S}$ be the space of all sentences, $X = (s_1, s_2)$ be the random variable representing a sample pair from $\mathcal{S}$, and Y the random variable representing the labels, with $Y = 1$ indicating that s_1 and s_2 are paraphrases and otherwise when $Y = 0$. We seek to separate the dependence between the distribution of positive and negative pairs, motivated by the observation of the presence of bias in the negative

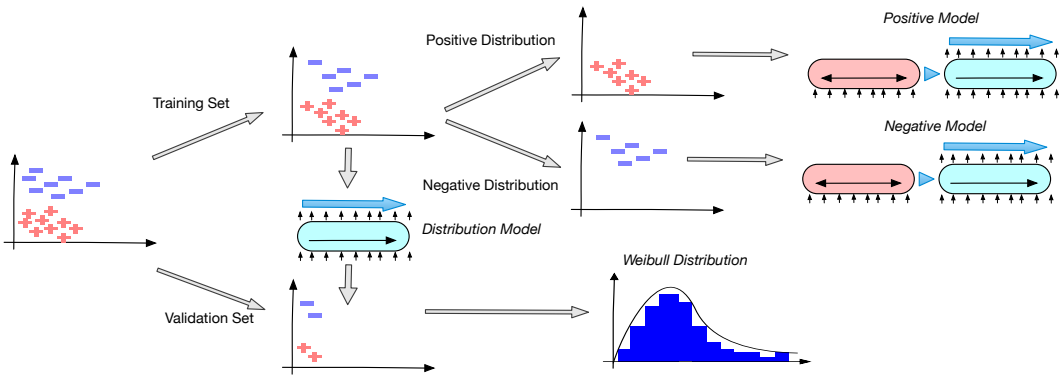

Figure 2: An overview of the training procedure of our model GAP. GAPX ensembles GAP with another discriminative model.

pairs. To begin, we model the distribution of sentences by splitting the sentence s_2 of length n into the autoregressive product of individual words, where $w_2^{(i)}$ denotes the ith word in s_2. By applying Bayesian Inference Law, we have:

$$P(Y=y|s_1,s_2) = \frac{P(Y=y|s_1)\Pi_{i=1}^{n} P(w_2^{(i)}|s_1, Y=y, w_2^{(1:i-1)})}{\Pi_{i=1}^{n} P(w_2^{(i)}|w_2^{(1:i-1)}, s_1)}. \tag{1}$$

Subtracting the logarithm for $Y=1$ and $Y=0$, we get:

$$\begin{aligned}
&\log(P(Y=1|s_1,s_2)) - \log(P(Y=0|s_1,s_2)) \\
&= (\log P(Y=1|s_1) - \log P(Y=0|s_1)) \\
&+ (\sum_{i=1}^{n} \log P(w_2^{(i)}|s_1, Y=1, w_2^{(1:i-1)}) - (\sum_{i=1}^{n} \log P(w_2^{(i)}|s_1, Y=0, w_2^{(1:i-1)})) \\
&= (\log P(Y=1) - \log P(Y=0)) \\
&+ (\sum_{i=1}^{n} \log P(w_2^{(i)}|s_1, Y=1, w_2^{(1:i-1)}) - (\sum_{i=1}^{n} \log P(w_2^{(i)}|s_1, Y=0, w_2^{(1:i-1)})).
\end{aligned} \tag{2}$$

In this way, we break the probability inference into 3 terms resulting in Eqn. 2: (1) $(\log P(Y=1) - \log P(Y=0))$, which should just be a constant; (2) $(\sum_{i=1}^{n} \log P(w_2^{(i)}|s_1, Y=1, w_2^{(1:i-1)})$, which depends only on the distribution of positive pairs; (3)$(\sum_{i=1}^{n} \log P(w_2^{(i)}|s_1, Y=0, w_2^{(1:i-1)})$, which depends only on the distribution of negative pairs. We define the score of confidence as follows:

$$S(s_1,s_2) = \underbrace{(\sum_{i=1}^{n} \log P(w_2^{(i)}|s_1, Y=1, w_2^{(1:i-1)})}_{\text{Positive Model}} - \underbrace{(\sum_{i=1}^{n} \log P(w_2^{(i)}|s_1, Y=0, w_2^{(1:i-1)}))}_{\text{Negative Model}}. \tag{3}$$

In the above, we are now left with two terms, the first representing the positive model and the second the negative model. If we were to train the two terms together, the effects of the negative pairs in the resulting model can never be removed during inference, which we have observed to be a major source of bias. To avoid this, we propose to train the first term and the second term separately, and then subsequently ensemble them based on the degree that a given pair is out of distribution. We train each model on top of the pretrained autoregressive transformer described in [30] known as BART. Given s_1 and s_2, we feed s_1 into the encoder as the condition, shift s_2 to the right by one-token, and feed shifted s_2 to the decoder. While the decoder proceeds autoregressively, we record the next-word probability distribution. We calculate the cross entropy between the next-word probability distribution and the target token in s_2 to update the model parameters. Note that here the Bayesian formulation has been similarly raised in some of the previous work like Moore and Lewis [37], but to the best of our knowledge, we're the first to propose this Bayesian formulation to control the reliance on different components of the model.

3.2 Ensembling

To combine the positive and negative model, if we know a priori $\mathcal{D}^t$ is in the same distribution as $\mathcal{D}^s$, we can directly substitute the prediction of the positive and negative model into Eqn. 3. We will refer to this as the In-distribution Predictor (IDP). If we have reason to believe that there is a significant distribution shift between $\mathcal{D}^s$ and $\mathcal{D}^t$ (e.g., different sources of corpus and different dataset collection procedure), we observe empirically that we should only utilize the positive model and disregard the negative model due to the bias it introduces. We will refer to this as the Out-of-distribution Predictor (OODP).

3.2.1 Automatic Ensembling

However, in most cases, we have little or no knowledge of the testing distribution, in which case we need to automatically decide how important the negative model is by detecting how much a test pair is in the same distribution as the training set. We adopt a weighted interpolation between a constant and the negative model in addition to the positive model as follow:

$$S(s_1, s_2) = \log P(s_2|s_1, Y = 1) - (1 - \lambda(s_1, s_2)) \log P(s_2|s_1, Y = 0) - \lambda(s_1, s_2)C, \tag{4}$$

where $\lambda(s_1, s_2)$ is a weight parameter depending on s_1 and s_2, and C is a constant that achieves a regularization effect. See Appendix for ablations on how C can be set. $P(s_2|s_1, Y = 1)$ and $P(s_2|s_1, Y = 0)$ are the same terms in Eqn.3. To automatically assign $\lambda(s_1, s_2)$ for different sentence pairs, we measure an out-of-distribution score for (s_1, s_2) with regard to the training distribution. Specifically, we use the same set of training data, comprising both positive and negative pairs, from $\mathcal{D}^s$, on which we train another autoregressive model, which we will refer to as the distribution model. The distribution model is trained by feeding an empty string into the encoder and the concatenated s_1 and s_2 into the decoder, with the training goal of predicting the next token. We measure the perplexity of each sentence pair (s_1, s_2) using the distribution model based on the following formula, w^i being the ith token of the concatenated (s_1, s_2) of length n:

$$PP(s_1, s_2) = \sqrt[n]{(\prod_{i=1}^{n} \frac{1}{P(w^i|w^{1:i-1})}}. \tag{5}$$

We then fit a Weibull distribution to the perplexity of a held-back set of validation data, so that it can better model the right-skewed property of the distribution. We derive the exponential parameter a, the shape parameter c, the location parameter loc, and the scale parameter $scale$. During testing, $\lambda(s_1, s_2)$ can now be determined as:

$$\lambda(s_1, s_2) = cdf(PP(s_1, s_2), Weibull(a, c, loc, scale)). \tag{6}$$

For the final prediction, we predict the sentence pair to be paraphrase if $S(s_1, s_2) \geq 0$ and non-paraphrase otherwise. This forms what we referred to earlier as GAP (Generalized Autoregressive Paraphrase-Identification).

3.2.2 Capturing Interplay Between Positive and Negative Pairs

In practice, training a positive and negative model separately disregards the interplay between the positive and negative pairs, which could be important when the test pairs are in-distribution. To capture such interplay, we utilize both positive and negative pairs to train a discriminative model for sequence classification. Specifically, we first define a thresholding function based on the value of λ:

$$\tau(\lambda) = \begin{cases} 0 & \lambda < 0.9 \\ 1 & else. \end{cases} \tag{7}$$

We then ensemble the discriminative model and GAP using the value of $\tau(\lambda)$:

$$S^*(s_1, s_2) = M(1 - \tau(\lambda(s_1, s_2)))(P(Y = 1|s_1, s_2) - \frac{1}{2}) + \tau(\lambda(s_1, s_2))S(s_1, s_2), \tag{8}$$

where $P(Y = 1|s_1, s_2)$ can be estimated by any discriminative model, and M is a sufficiently large constant. Note that this definition is essentially the same as trusting the discriminative model when we do not have statistical evidence that the pair is out-of-distribution (p-test < 10%) while trusting the GAP model otherwise. For the final prediction, we predict the sentence pair to be paraphrase if $S * (s_1, s_2) \geq 0$ and non-paraphrase otherwise. This defines GAPX (Generalized Autoregressive Paraphrase Identfication X), for which we set M to be sufficiently large (> 1000), so that when comparing the model confidence for different pairs, the score of the discriminative model will be prioritized.

4 Experiments

Our experiments are designed to (1) verify that the task of paraphrase identification suffers from biases in the datasets that is the main obstacle to generalization in this field of study, (2) test the accuracy of our perplexity based out-of-distribution detection method, and (3) test that balancing the utilization of the negative model can help outperform the state-of-the-art in the face of distribution shift, without losing in the in-distribution scenarios.

4.1 Datasets

We compare our method against the other state-of-the-art methods on different combinations of the following datasets:

- Quora Question Pair (QQP) consists of over 400,000 lines of potential question duplicate pairs. Since the original sampling method returns an imbalanced dataset, the authors attempt to balance it with additional negative pairs collected from similar topics to make them harder. Note that to scale QQP down to approximately the same size of PAWS and PIT (see below), we take the first 10k training pairs and 2k testing pairs from the train and test split by Wang et al. [52].
- World Machine Translation Metrics Task 2017 (WMT) [8] contains in total 3793 manual ratings of machine translations from 7 languages to English. Each rating result contains a source sentence in the source language, a reference sentence in English (ground truth translation), a machine translated sentence in English, and a manual rating of the quality of translations. We take the ground-truth reference sentence and the machine translated sentence as the sentence pair. Sentences with a higher quality score (> 0) are labeled as paraphrases, while those with lower quality scores (≤ 0) are labeled otherwise. The resulting paraphase identification dataset is balanced. Note that this dataset is significantly smaller than other datasets, so we only use it as a test set.
- Paraphrase and Semantic Similarity in Twitter (PIT) [53, 54] contains 18762 sentence pairs automatically extracted from a similar distribution of topics as QQP. Annotators manually assigned integer scores from 0 to 5 to each sentence pair, representing the degree of similarity between the sentence pair. To make it a paraphrase identification dataset, we label sentence pairs with low scores (0, 1) as non-paraphrases and those with high scores (4,5) as paraphrases. The original dataset is unbalanced, so we randomly sample a maximum balanced subset of the dataset. The original test set processed in this way shrinks to only 350, and is not comparable to the other datasets. Hence, we use the original development data of size 1896 as the test set while keeping original test set of size 350 for development. The training set contains 5332 sentence pairs.
- Paraphrase Adversarials from Word Scrambling (PAWS) [62] contains 49,401 sentence pairs, each of which is constructed from the same bag-of-word to make the evaluation more challenging. Most of the negative pairs are generated by word swapping while positive pairs are supplemented by back translation. This dataset contains paraphrase pairs that are the adversarial counterparts of standard paraphrase datasets such as QQP and PIT.

With these datasets, we perform experiments where different models (Sec. 4.2) are trained on one dataset and evaluated on another in order to observe whether their performance hold in the face of distribution shift.

4.2 Benchmarks

We benchmark the paraphrase identification task with these methods:

- **BOW** [62] represents two input sentences with bag of words. The bag of words representation of each input sentence is passed through a fully-connected network and cosine similarity between of the final layer is used to compute the classification output.
- **BiLSTM** [28] passes each of the two input sentences through a bidirectional LSTM network. The output state of the two sentences are then concatenated together and passed through a fully-connected network to get the classification output.
- **BERT** [16] is representative of the state-of-the-art transformer methods for text classification. We finetune the pretrained model "bert-base-uncased" in a standard way. We concatenate the sentence pair separated by a [SEP] token and take the [CLS] token as aggregate representation for the sentence pair. The embedding for the [CLS] token is then fed into an output layer for classification.
- **BART** [30] is the original transformer model that we build on by finetuning the pretrained model "bart-base-uncased". We concatenate the sentence pair separated by a </s> token, feeding it both into the encoder and the decoder. We use the <s> token at the end of the sentence pair for aggregate representation so that it can attend to decoder states from the complete input. The embedding for <s> is then fed into an output layer for classification.
- **RoBERTa** [34] shares the same transformer architecture with BERT, but uses a more robust pretraining strategy, and as a result performs better than BERT in many NLP tasks [34]. For our experiments, we employ RoBERTa in the same way as BERT.
- **IDP** (In-distribution Predictor) is our model for in-distribution prediction if we know a priori that the testing pairs come from the same distribution as the training pairs. It combines the positive and negative models as given in Eqn. 3.
- **OODP** (Out-of-distribution Predictor) is our model for out-of-distribution prediction if we know a priori that the testing and training pairs are not in the same distribution. It only makes use of the positive model. We expect our OODP to have better generalizability than our IDP, because of its reduced reliance on negative examples.
- **GAP** (Generalized Autoregressive Paraphrase-Identification) is our method that utilizes the perplexity based out-of-distribution detector to automatically control the reliance on the negative model, using Eqn. 4. This setting is different from IDP and OODP in that we do not have a priori knowledge of the test distribution.
- **GAPX** (Generalized Autoregressive Paraphrase-Identification X) ensembles GAP with RoBERTa described above, because we found RoBERTa to be the strongest baseline for in-distribution prediction. The intention is to capture via RoBERTa the interplay between positive and negative pairs. As depicted in Eqn. 8, when we do not have significant evidence that the given pair is out of distribution (p-test > 10%), we trust the prediction given by the discriminative model (RoBERTa). Otherwise, we trust GAP.

Following the previous literature in sentence matching [52, 16], we mainly use macro F1 score (F1) and accuracy score (ACC) to evaluate the models. Results based on Area-under-curve of the Receiver Operating Characteristic Curve (AUROC), a common metrics used to evaluate out-of-distribution metrics, are also provided in Appendix.A.

4.3 Measuring Distribution Shift

To situate our experiments properly, we note that different datasets does not equate to different distributions. It is thus important for us to be able to measure the distribution gap between datasets, and shed light on the models that perform the best when transferring between datasets with high distribution gap. Metrics that is per sample based such as our perplexity measure are not suitable for measuring at dataset level. To this end, we take a look at the Reverse Classification Accuracy* (RCA*) metric that has been proposed to predict the drop of model performance [18] and model selection [19, 64]. Here, given D^s and D^t, we first train a model $M1$ from the training set of D^s. We then take a certain amount of samples from the target distribution D^t (1000 in our experiments), and use $M1$ to relabel them. The relabeled pairs are then utilized to train a new model $M2$. We measure

the performance (in terms of AUC or ACC) of $M2$ on a held-out test set from D^s. As a control, we train another model $M3$ following the same procedure except the relabeled data comes from D^s. We denote the performance drop from $M3$ to $M2$ as the RCA* score indicative of the distribution gap.

To calibrate RCA*, for each dataset, we randomly selected 1000 held-out pairs and measure its shift from the dataset itself (which we expect to characterize an in-distribution RCA*). After 100 repetitions of measurements, we get a probability distribution of RCA* scores for each distribution in itself. Calibration results are reported in Appendix. 9. All the distributions share a mean of around 0 and a standard deviation around 2. By a p-test of 10%, it is unlikely that datasets with a RCA* greater than 4.0 would belong to the same distribution. Based on this observation, most of the transfers between different datasets are likely out-of-distribution, except for transfers from QQP to PIT and vice versa, which have a RCA* score of 2.8 and 3.4 respectively. This is probably because QQP and PIT are curated in a similar fashion, where they extract sentences with similar topics from social platforms, while all other datasets adopt different strategies to collect their data.

Finally, it is important to note that although RCA* provides a good estimate of the distribution shift at the dataset level, it's utility does not easily extend to Eqn. 4 and 8 as opposed to per sample out-of-distribution metrics. RCA* assumes availability of the entire test set, while in real world, we are much more likely to get a single or small batches of test pairs at a time, all of which could even be from different distributions.

4.4 Implementation details

In practice, for testing, it helps to average the cross entropy by the length of the sentence, and average the cross entropy of generating s_2 from s_1 and generating s_1 from s_2. To optimize conditional sentence generators, we use Adam optimizer with learning rate 2e-5. We adopt cross entropy loss for each word logit. All experiments are run on Nvidia 2080 Ti with 11 GB memory.

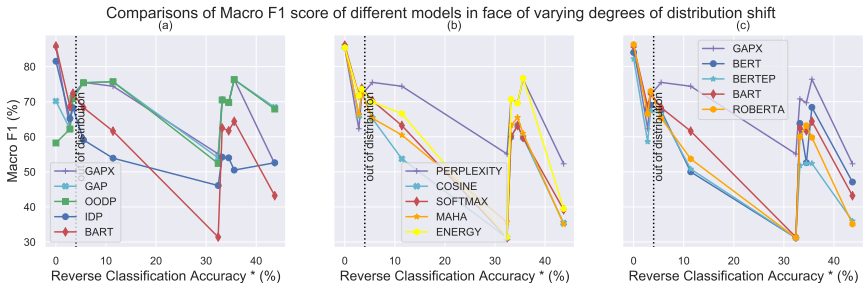

Figure 3: Comparing Macro F1 scores of different models at varying degrees of distribution shift.

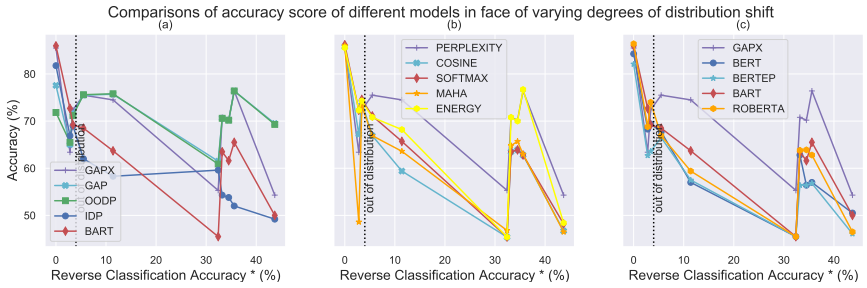

Figure 4: Comparing ACC score of different models at varying degrees of distribution shift.

4.5 Main Results

Bias in Negative Pairs To understand whether negative pairs are major sources of bias, we plot the Macro F1 and ACC score against RCA* in Fig. 3(a) and Fig. 4(a), comparing the performances of OODP, IDP, and BART, all finetuned from the same pretrained checkpoint. The x-axis is plotted in ascending order of RCA* between pairs of datasets given in Table 1 and 2. There are three pairs

Model	QQP -> WMT (5.5)	PIT -> WMT (11.4)	PIT -> PAWS (32.4)	PAWS -> QQP (33.2)	PAWS -> PIT (34.5)	PAWS -> WMT (35.6)	QQP -> PAWS (43.7)	average
BOW	34.6/51.5	33.3/51.4	35.3/54.7	33.3/50.0	34.3/50.2	34.8/51.7	35.3/54.7	34.4/52.0
BiLSTM	34.4/51.5	50.7/51.1	48.6/48.7	36.8/50.4	43.3/50.6	34.9/51.2	37.1/54.7	40.8/51.2
BERT	67.4/67.7	50.0/57.7	31.2/45.5	63.8 /62.8	52.6/56.4	68.4/57.0	47.1/50.5	54.4/56.8
BERT+EP	66.5/66.5	50.8/57.5	31.2/45.5	51.8/56.4	52.4/56.4	52.4/56.8	36.0/46.1	48.7/55.0
BART	68.3/68.5	61.6/63.7	31.4/45.5	62.5/63.5	61.7/61.6	64.4/65.5	43.2/50.0	56.2/59.8
RoBERTa	65.3/66.9	53.7/59.4	31.2/45.5	60.0/63.6	63.2/63.9	59.8/62.8	35.1/46.5	52.6/58.4
IDP	59.1/62.0	53.9/58.3	46.1/59.6	54.2 /54.3	54.0/53.8	50.5/52.0	52.6/49.2	52.9/55.6
OODP	75.4/**75.6**	**75.7/75.8**	52.4/60.9	70.5/70.6	**69.8/70.2**	76.3/**76.4**	67.9/69.3	69.7/**71.3**
GAP	75.4/75.5	**75.7**/75.7	54.0/**61.5**	70.5/70.6	69.7/**70.2**	76.3/**76.4**	**68.4/69.5**	**70.0/71.3**
GAPX	**75.5**/75.5	74.4/74.5	**55.1**/55.5	**70.8/70.8**	69.7/**70.2**	**76.4/76.4**	52.3/54.3	67.7/68.2

Table 1: Model performance on different out-of-distribution combinations of QQP, PAWS and PIT, in terms of macro F1/accuracy (ACC). Parenthesized is the RCA* score for each combination of datasets.

Model	QQP ->QQP (0)	PAWS ->PAWS (0)	PIT ->PIT (0)	QQP ->PIT (2.8)	PIT ->QQP (3.4)	average
BOW	51.3/57.8	48.8/56.8	33.3/50.0	41.7/49.7	33.3/50.0	40.7/52.9
BiLSTM	61.6/63.6	43.6/53.0	50.6/51.1	51.7/52.9	41.1/49.0	49.7/53.9
BERT	82.5/82.6	92.7/93.2	76.9/77.0	68.0/68.3	69.0/69.4	77.8/78.1
BERT+EP	81.6/81.7	89.7/89.7	75.3/74.9	58.6/62.7	67.4/63.6	74.5/74.5
BART	82.6/82.8	94.1/94.1	80.9/81.0	68.6/72.7	72.4/69.2	79.7/80.0
RoBERTa	84.4/84.5	93.5/93.6	81.0/81.1	66.5/68.8	73.0/74.9	79.7/80.6
OODP	65.3/73.2	67.9/77.1	41.5/65.2	62.2/65.5	65.2/71.2	60.4/70.4
IDP	79.0/79.1	88.2/88.5	77.4/77.7	65.2/66.9	68.3/69.0	75.6/76.2
GAP	68.8/71.0	85.1/85.2	56.6/76.5	62.2/65.0	71.3/71.7	68.8/73.9
GAPX	84.4/84.5	92.7/92.7	79.3/79.3	62.3/63.4	72.0/72.4	78.1/78.5

Table 2: Model performance for in-distribution performances on QQP, PAWS, and PIT, in terms of macro F1/accuracy (ACC). Parenthesized is the RCA* score for each combination of datasets.

that have RCA* score of 0 (Table 2), for which we average the performance in the plots. Both the F1 and ACC plots share a similar pattern. In the in-distribution region, where the RCA* score is less than 4% (Fig. 9), BART and IDP achieves similarly high performances of 79.7% and 75.6% F1 on average respectively. The 4.1% gap in F1 is potentially due to the fact that IDP trains a negative model and positive model separately, neglecting the interaction between positive and negative pairs. Comparatively, the performance of OODP is significantly inferior to the other two, with only 60.4% average F1. This changes in the out-of-distribution region, where the RCA* score is now greater than 4%. OODP turns out to be the leading model over BART and IDP. With increasing degree of distribution shift, we observe that both BART and IDP are especially fragile and their performance drop significantly. When the RCA* is greater than 20, both models' F1 drop to as low as around 60%, which can be hardly useful in practice. In contrast, OODP maintains an advantage in F1 of as much as 10-20% throughout the out-of-distribution region. Since the only difference between OODP and IDP is that OODP transfers only the positive model, it confirms our hypothesis that the negative model does not generalize as well as the positive model.

Importance of the Interplay between Positive and Negative Pairs To understand the necessity of capturing the interplay for test pairs that are in distribution, we compare the performances of GAP, GAPX, and BART in Fig. 3(a) and Fig. 4(a). GAP only ensembles the positive and negative model trained separately, so does not contain any interplay information. On the other hand, both BART, trained with positive and negative pairs together, and GAPX (Sec. 3.2.2) capture interplay information. As shown in the plot, the performance of GAP in the in-distribution region is not directly comparable to BART with a gap of 10.9% in macro F1. In contrast, GAPX's in-distribution result has a much smaller margin of 1.6% macro F1 compared to BART, yet, by automatically weighing the contribution of the discriminative model, GAPX also closely matches the performance of GAP in the out-of-distribution region with only a 2.3% loss in macro F1.

Effectiveness of Perplexity-based Ensembling We also substitute $PP(s_1, s_2)$ with other state-of-the-art out-of-distribution metrics used for estimating the probability in Eqn. 6 for GAPX. Specifically, we compare our perplexity metric with Maximum Softmax Probability (SOFTMAX) [24, 26, 6], Energy Score (ENERGY) [33], Mahalanobis Distance (MAHA) [29], and COSINE [65]. See Appendix for more details. The results are given in Fig. 4(b) and Fig. 3(b). F1 and ACC are all similar in the in-distribution region. However, in the out-of-distribution region, SOFTMAX, MAHA, and COSINE start to perform poorly. ENERGY turned out to be the most robust but is still obviously not matching our perplexity metric, often by a large margin.

Generalization We implemented one of the most popular methods for domain generalization in paraphrase identification, Expert Product [15], as a potential strong baseline for handling dataset bias. As described in Sec. 4.2, we train a BERT classifier with only the first sentence s_1 as the biased model for BERT+EP. We report the results in Fig. 4(c) and Fig. 5(c), together with BERT, BART and RoBERTa. In addition, we also provide performance of traditional methods like BoW [62] and ESIM [12] in Table 1 and Table 2. BERT, BART, and RoBERTa all produce similar results on all combinations of datasets. We observe that their performances are consistently better than traditional methods like BoW and ESIM, showing that pretraining and finetuning can indeed improve the generalizability of classifiers for paraphrase identification. However, their performances in out-of-distribution setting are still far from their in-distribution performances, with accuracy below 65% in most of the cases (around 20% drop). GAPX maintains an absolute margin of around 10% in terms of ACC and an absolute margin of 7-20% in terms of F1 in the out-of-distribution region. The best transformer-based models in the out-of-distribution region is BART, with an average of 56.2% in F1 and 59.8% in ACC, while GAPX maintains an average of 67.7% in F1 and 68.2% in ACC, with an absolute gain of 11.5% in F1 and 8.4% in ACC. What is also encouraging is that GAPX's performance is close to that of OODP, which is promising that we do not need *a priori* information on the domain gap between the source and target. Lastly, BERT+EP fails to provide much gain, which we conjecture is due to the difficulty of "finding" the right bias model or features.

5 Conclusion

We have shown that negative samples introduce bias that prevent the generalization of paraphrase identification models. To overcome, we present a novel paradigm to train separate models for the distribution of positive and negative samples independently, and utilize a perplexity based out-of-distribution detection to ensemble them automatically. Experiments show that our method achieves an average of 11.5% gain of F1 and 8.4% gain of ACC in various different out-of-distribution scenarios over other state-of-the-art methods.

5.1 Limitations

Our methodology is specifically designed for only "verification" problems, where the samples come in pairs. Scenarios that involve other types of bias will require non-trivial turn-key formulations to explicitly model the source of bias (much like the negative model).

6 Acknowledgements

This work is sponsored by Meta AI. We would also like to thank Prof. Yoav Artzi for a helpful discussion in the early stage of this project.

References

[1] D. Abati, A. Porrello, S. Calderara, and R. Cucchiara. Latent space autoregression for novelty detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 481–490, 2019.

[2] A. Agrawal, D. Batra, D. Parikh, and A. Kembhavi. Don't just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4971–4980, 2018.

[3] H. Amirkhani, M. AzariJafari, Z. Pourjafari, S. Faridan-Jahromi, Z. Kouhkan, and A. Amirak. Farstail: A persian natural language inference dataset. *arXiv preprint arXiv:2009.08820*, 2020.

[4] A. Anand, E. Belilovsky, K. Kastner, H. Larochelle, and A. C. Courville. Blindfold baselines for embodied QA. *CoRR*, abs/1811.05013, 2018. URL `http://arxiv.org/abs/1811.05013`.

[5] H. Bahng, S. Chun, S. Yun, J. Choo, and S. J. Oh. Learning de-biased representations with biased representations. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 528–539. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/bahng20a.html`.

[6] L. Bergman and Y. Hoshen. Classification-based anomaly detection for general data. *ArXiv*, abs/2005.02359, 2020.

[7] R. Bhagat and E. Hovy. What Is a Paraphrase? *Computational Linguistics*, 39(3):463–472, 09 2013. ISSN 0891-2017. doi: 10.1162/COLI_a_00166. URL `https://doi.org/10.1162/COLI_a_00166`.

[8] O. Bojar, Y. Graham, and A. Kamran. Results of the wmt17 metrics shared task. In *WMT*, 2017.

[9] S. Bulusu, B. Kailkhura, B. Li, P. K. Varshney, and D. Song. Anomalous example detection in deep learning: A survey. *IEEE Access*, 8:132330–132347, 2020. doi: 10.1109/ACCESS.2020.3010274.

[10] R. Cadène, C. Dancette, H. Ben-younes, M. Cord, and D. Parikh. Rubi: Reducing unimodal biases in visual question answering. *CoRR*, abs/1906.10169, 2019. URL `http://arxiv.org/abs/1906.10169`.

[11] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada, Aug. 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2001. URL `https://aclanthology.org/S17-2001`.

[12] Q. Chen, X. Zhu, Z. Ling, S. Wei, and H. Jiang. Enhancing and combining sequential and tree LSTM for natural language inference. *CoRR*, abs/1609.06038, 2016. URL `http://arxiv.org/abs/1609.06038`.

[13] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020. URL `https://arxiv.org/abs/2002.05709`.

[14] H. Choi, E. Jang, and A. A. Alemi. Waic, but why? generative ensembles for robust anomaly detection, 2019.

[15] C. Clark, M. Yatskar, and L. Zettlemoyer. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. *CoRR*, abs/1909.03683, 2019. URL `http://arxiv.org/abs/1909.03683`.

[16] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

[17] T. DeVries and G. W. Taylor. Learning confidence for out-of-distribution detection in neural networks, 2018. URL `https://arxiv.org/abs/1802.04865`.

[18] H. Elsahar and M. 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, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1222. URL `https://aclanthology.org/D19-1222`.

[19] W. Fan and I. Davidson. Reverse testing: an efficient framework to select amongst classifiers under sample selection bias. In *KDD '06*, 2006.

[20] S. Fort, J. Ren, and B. Lakshminarayanan. Exploring the limits of out-of-distribution detection. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 7068–7081. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/3941c4358616274ac2436eacf67fae05-Paper.pdf.

[21] J. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. Á. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent: A new approach to self-supervised learning. *CoRR*, abs/2006.07733, 2020. URL https://arxiv.org/abs/2006.07733.

[22] H. He, S. Zha, and H. Wang. Unlearn dataset bias in natural language inference by fitting the residual. *CoRR*, abs/1908.10763, 2019. URL http://arxiv.org/abs/1908.10763.

[23] H. He, S. Zha, and H. Wang. Unlearn dataset bias in natural language inference by fitting the residual. *arXiv preprint arXiv:1908.10763*, 2019.

[24] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016. URL http://arxiv.org/abs/1610.02136.

[25] D. Hendrycks, M. Mazeika, and T. G. Dietterich. Deep anomaly detection with outlier exposure. *CoRR*, abs/1812.04606, 2018. URL http://arxiv.org/abs/1812.04606.

[26] Y. Hsu, Y. Shen, H. Jin, and Z. Kira. Generalized ODIN: detecting out-of-distribution image without learning from out-of-distribution data. *CoRR*, abs/2002.11297, 2020. URL https://arxiv.org/abs/2002.11297.

[27] Z. Kozareva and A. Montoyo. Paraphrase identification on the basis of supervised machine learning techniques. In *International conference on natural language processing (in Finland)*, pages 524–533. Springer, 2006.

[28] W. Lan and W. Xu. Neural network models for paraphrase identification, semantic textual similarity, natural language inference, and question answering. *CoRR*, abs/1806.04330, 2018. URL http://arxiv.org/abs/1806.04330.

[29] K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks, 2018. URL https://arxiv.org/abs/1807.03888.

[30] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019. URL http://arxiv.org/abs/1910.13461.

[31] B. Li, T. Liu, B. Wang, and L. Wang. Enhancing deep paraphrase identification via leveraging word alignment information. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7843–7847, 2021. doi: 10.1109/ICASSP39728.2021.9414944.

[32] S. Liang, Y. Li, and R. Srikant. Principled detection of out-of-distribution examples in neural networks. *CoRR*, abs/1706.02690, 2017. URL http://arxiv.org/abs/1706.02690.

[33] W. Liu, X. Wang, J. D. Owens, and Y. Li. Energy-based out-of-distribution detection. *CoRR*, abs/2010.03759, 2020. URL https://arxiv.org/abs/2010.03759.

[34] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

[35] E. Marsi and E. Krahmer. Explorations in sentence fusion. 01 2005.

[36] S. Min, E. Wallace, S. Singh, M. Gardner, H. Hajishirzi, and L. Zettlemoyer. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4249–4257, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1416. URL https://aclanthology.org/P19-1416.

[37] R. C. Moore and W. Lewis. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL https://aclanthology.org/P10-2041.

[38] S. Pidhorskyi, R. Almohsen, and G. Doretto. Generative probabilistic novelty detection with adversarial autoencoders. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/5421e013565f7f1afa0cfe8ad87a99ab-Paper.pdf.

[39] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[40] R. Rei, C. Stewart, A. C. Farinha, and A. Lavie. COMET: A neural framework for MT evaluation. *CoRR*, abs/2009.09025, 2020. URL https://arxiv.org/abs/2009.09025.

[41] R. Rei, C. Stewart, A. C. Farinha, and A. Lavie. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.213. URL https://aclanthology.org/2020.emnlp-main.213.

[42] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084, 2019. URL http://arxiv.org/abs/1908.10084.

[43] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/1e79596878b2320cac26dd792a6c51c9-Paper.pdf.

[44] T. Sellam, D. Das, and A. P. Parikh. BLEURT: learning robust metrics for text generation. *CoRR*, abs/2004.04696, 2020. URL https://arxiv.org/abs/2004.04696.

[45] X. Shen and W. Lam. Towards domain-generalizable paraphrase identification by avoiding the shortcut learning. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 1318–1325, Held Online, Sept. 2021. INCOMA Ltd. URL https://aclanthology.org/2021.ranlp-main.148.

[46] B. Thompson and M. Post. Automatic machine translation evaluation in many languages via zero-shot paraphrasing. *CoRR*, abs/2004.14564, 2020. URL https://arxiv.org/abs/2004.14564.

[47] G. S. Tomar, T. Duque, O. Täckström, J. Uszkoreit, and D. Das. Neural paraphrase identification of questions with noisy pretraining. *arXiv preprint arXiv:1704.04565*, 2017.

[48] P. A. Utama, N. S. Moosavi, and I. Gurevych. Towards debiasing NLU models from unknown biases. *CoRR*, abs/2009.12303, 2020. URL https://arxiv.org/abs/2009.12303.

[49] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

[50] S. Wang, H. Fang, M. Khabsa, H. Mao, and H. Ma. Entailment as few-shot learner. *CoRR*, abs/2104.14690, 2021. URL https://arxiv.org/abs/2104.14690.

[51] Z. Wang, H. Mi, and A. Ittycheriah. Sentence similarity learning by lexical decomposition and composition. *CoRR*, abs/1602.07019, 2016. URL http://arxiv.org/abs/1602.07019.

[52] Z. Wang, W. Hamza, and R. Florian. Bilateral multi-perspective matching for natural language sentences. *CoRR*, abs/1702.03814, 2017. URL http://arxiv.org/abs/1702.03814.

[53] W. Xu, A. Ritter, C. Callison-Burch, W. B. Dolan, and Y. Ji. Extracting lexically divergent paraphrases from Twitter. *Transactions of the Association for Computational Linguistics*, 2: 435–448, 2014. doi: 10.1162/tacl_a_00194. URL https://www.aclweb.org/anthology/Q14-1034.

[54] W. Xu, C. Callison-Burch, and B. Dolan. SemEval-2015 Task 1: Paraphrase and semantic similarity in Twitter (PIT). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 1–11. Association for Computational Linguistics, 2015. doi: 10.18653/v1/S15-2001. URL https://www.aclweb.org/anthology/S15-2001.

[55] J. Yang, K. Zhou, Y. Li, and Z. Liu. Generalized out-of-distribution detection: A survey. *CoRR*, abs/2110.11334, 2021. URL https://arxiv.org/abs/2110.11334.

[56] Y. Yang, Y. Zhang, C. Tar, and J. Baldridge. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. *CoRR*, abs/1908.11828, 2019. URL http://arxiv.org/abs/1908.11828.

[57] W. Yin and H. Schütze. Convolutional neural network for paraphrase identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 901–911, 2015.

[58] W. Yuan, G. Neubig, and P. Liu. Bartscore: Evaluating generated text as text generation. *CoRR*, abs/2106.11520, 2021. URL https://arxiv.org/abs/2106.11520.

[59] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny. Barlow twins: Self-supervised learning via redundancy reduction. *CoRR*, abs/2103.03230, 2021. URL https://arxiv.org/abs/2103.03230.

[60] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang. Deep structured energy based models for anomaly detection. *CoRR*, abs/1605.07717, 2016. URL http://arxiv.org/abs/1605.07717.

[61] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with BERT. *CoRR*, abs/1904.09675, 2019. URL http://arxiv.org/abs/1904.09675.

[62] Y. Zhang, J. Baldridge, and L. He. PAWS: paraphrase adversaries from word scrambling. *CoRR*, abs/1904.01130, 2019. URL http://arxiv.org/abs/1904.01130.

[63] W. Zhao, M. Peyrard, F. Liu, Y. Gao, C. M. Meyer, and S. Eger. Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance. *CoRR*, abs/1909.02622, 2019. URL http://arxiv.org/abs/1909.02622.

[64] E. Zhong, W. Fan, Q. Yang, O. Verscheure, and J. Ren. Cross validation framework to choose amongst models and datasets for transfer learning. pages 547–562, 09 2010. ISBN 978-3-642-15938-1. doi: 10.1007/978-3-642-15939-8_35.

[65] W. Zhou and M. Chen. Contrastive out-of-distribution detection for pretrained transformers. *CoRR*, abs/2104.08812, 2021. URL https://arxiv.org/abs/2104.08812.

[66] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] . See our listed contributions in the introduction.

(b) Did you describe the limitations of your work? [Yes] . See the end of our conclusion, Sec. 5.1.

(c) Did you discuss any potential negative societal impacts of your work? [N/A] . This is a traditional Natural Language Processing research topic. We do not see any conceivable negative social impact.

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [Yes] . See Sec. 3.1

(b) Did you include complete proofs of all theoretical results? [Yes] . See Section. 3.1

3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] .

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] . See Sec.4.4

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] The baseline models in NLP that we are comparing to do not report error bars in their paper.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] . See Sec. 4.4.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] . See Appendix 4.1

(b) Did you mention the license of the assets? [N/A] . Those are all public datasets.

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] . Those are all public datasets, allowing researching purpose.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] . Those data does not contain personal information

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A AUROC of Main Results

We include full results of AUROC scores here.

Model	QQP -> WMT (5.5)	PIT -> WMT (11.4)	PIT -> PAWS (32.4)	PAWS -> QQP (33.2)	PAWS -> PIT (34.5)	PAWS -> WMT (35.6)	QQP -> PAWS (43.7)	average
BOW	51.9	52.5	49.0	56.1	44.0	53.2	49.1	50.8
BiLSTM	50.1	51.5	49.3	50.4	50.6	49.8	50.1	50.3
BERT	75.0	70.7	51.2	69.2	58.2	70.7	54.7	64.2
BERT+EP	73.5	74.6	52.2	66.9	62.3	69.4	53.4	64.6
BART	75.7	76.4	53.3	71.7	65.0	77.6	56.6	68.0
RoBERTa	77.9	76.9	52.4	77.4	71.2	80.6	54.3	70.1
IDP	71.0	65.7	63.9	56.0	55.0	50.3	67.0	61.3
OODP	85.0	**85.0**	**67.2**	76.7	**77.5**	**84.9**	74.4	**78.7**
GAP	**85.1**	84.7	66.4	76.2	**77.5**	**84.9**	**74.7**	78.5
GAPX	83.8	81.1	58.3	**77.7**	77.5	84.6	59.5	74.6

Table 3: Model performance on different out-of-distribution combinations of QQP, PAWS and PIT, in terms of area under curve (AUROC).

Model	QQP ->QQP (0)	PAWS ->PAWS (0)	PIT ->PIT (0)	QQP ->PIT (2.8)	PIT ->QQP (3.4)	average
BOW	60.9	57.7	47.7	47.4	61.3	55.0
BiLSTM	64.6	49.6	50.9	52.8	49.0	53.4
BERT	90.0	97.6	85.1	77.3	73.4	84.7
BERT+EP	89.8	95.8	82.7	76.1	73.7	83.6
BART	90.6	98.2	89.1	78.2	77.9	86.8
RoBERTa	92.0	98.7	90.2	82.0	80.0	88.6
OODP	79.6	84.6	73.6	72.8	76.4	77.4
IDP	88.1	95.4	86.8	77.9	74.6	84.6
GAP	78.9	92.3	84.5	72.3	76.4	80.9
GAPX	90.7	98.1	87.4	72.5	79.2	85.6

Table 4: Model performance for in-distribution performances on QQP, PAWS, and PIT, in terms of area under curve (AUROC).

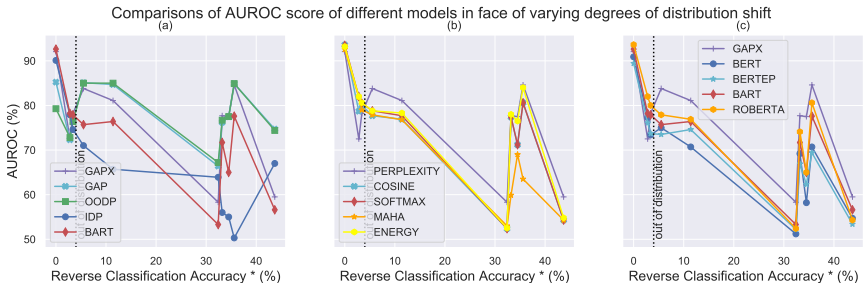

Figure 5: Comparing AUROC scores of different models at varying degrees of distribution shift.

B Additional Ablations

B.1 Additional Variants

We also conducted ablations on several variants of GAPX.

- **GAPX(neg-log)** replaces perplexity with the neg-log likelihood of the concatenated (s_1, s_2) as the measure. Specifically, GAPX(neg-log) modifies the Eqn. 5 and Eqn. 6 such that:

$$NLL(s_1, s_2) = \log((\prod_{i=1}^{n} \frac{1}{P(w^i|w^{1:i-1})}).$$

$$\lambda(s_1, s_2) = cdf(NLL(s_1, s_2), Weibull(a, c, loc, scale)).$$

- **GAPX(w/ IDP)** ensembles GAP with IDP.
- **GAPX(w/ BART)** explores the option of ensembling GAP with BART as described in Appendix.4.2.

These variants of GAPX mostly perform similarly to the GAPX model proposed in the main paper, which ensembles GAP with RoBERTa.

Model	QQP -> WMT (5.5)	PIT -> WMT (11.4)	PIT -> PAWS (32.4)	PAWS -> QQP (33.2)	PAWS -> PIT (34.5)	PAWS -> WMT (35.6)	QQP -> PAWS (43.7)	average
GAPX	75.5/75.5	74.4/74.5	55.1/55.5	70.8/70.8	69.7/70.2	76.4/76.4	52.3/54.3	67.7/68.2
GAPX(neg-log)	75.9/75.9	74.8/74.8	56.0/61.7	62.6/65.9	62.9/63.4	72.3/72.4	62.9/63.0	66.8/68.2
GAP(w/ IDP)	73.2/73.4	74.0/74.0	55.4/61.6	57.8/58.3	54.7/54.7	67.7/67.9	66.3/66.5	64.2/65.2
GAPX(w/ BART)	74.6/74.6	74.6/74.6	55.2/55.4	70.7/70.8	69.7/70.2	75.9/76.0	55.1/56.0	68.0/68.2
GAPX(w/ BERT)	72.6/72.6	74.7/74.7	55.1/55.2	70.2/70.2	69.7/70.2	75.7/75.9	56.1/56.3	67.7/67.9

Table 5: Performance on different out-of-distribution combinations of QQP, PAWS and PIT, in terms of macro F1/accuracy (ACC).

Model	QQP ->QQP (0)	PAWS ->PAWS (0)	PIT ->PIT (0)	QQP ->PIT (2.8)	PIT ->QQP (3.4)	average
GAPX	84.4/84.5	92.7/92.7	79.3/79.3	62.3/63.4	72.0/72.4	78.1/78.5
GAPX(neg-log)	83.3/83.4	91.2/91.5	80.9/80.9	68.9/70.1	72.2/72.7	79.3/78.7
GAP(threshed)	78.2/78.3	86.5/86.8	77.4/77.7	67.4/68.5	69.4/69.8	75.8/76.2
GAPX(w/ BART)	82.6/82.8	93.0/93.1	79.4/79.4	62.3/63.4	71.4/71.5	77.7/78.0
GAPX(w/ BERT)	82.6/82.7	91.9/92.0	76.8/76.8	62.3/63.4	69.9/69.9	76.7/77.0

Table 6: In-distribution performances on QQP, PAWS, and PIT, in terms of macro F1/accuracy (ACC).

Model	QQP -> WMT (5.5)	PIT -> WMT (11.4)	PIT -> PAWS (32.4)	PAWS -> QQP (33.2)	PAWS -> PIT (34.5)	PAWS -> WMT (35.6)	QQP -> PAWS (43.7)	average
GAPX(perplexity)	75.5/75.5	74.4/74.5	55.1/55.5	70.8/70.8	69.7/70.2	76.4/76.4	52.3/54.3	67.7/68.2
GAPX(cosine)	65.2/66.9	53.7/59.4	31.2/45.4	60.0/63.5	63.2/63.9	59.8/62.8	35.4/46.7	52.7/58.4
GAPX(softmax)	70.3/71.1	63.2/65.7	31.2/45.4	60.0/63.5	63.2/63.9	59.8/62.8	39.2/48.2	55.3/60.1
GAPX(maha)	65.3/66.9	60.5/63.6	35.7/46.8	63.2/64.8	65.5/65.7	61.0/63.0	35.1/46.5	55.2/59.6
GAPX(energy)	69.9/70.8	66.6/68.2	31.4/45.4	70.7/70.8	69.5/70.0	76.7/76.7	39.5/48.4	60.6/64.3

Table 7: Out-of-distribution performance when using different out-of-distribution metrics.

Model	QQP ->QQP (0)	PAWS ->PAWS (0)	PIT ->PIT (0)	QQP ->PIT (2.8)	PIT ->QQP (3.4)	average
GAPX(perplexity)	84.4/84.5	92.7/92.7	79.3/79.3	62.3/63.4	72.0/72.4	78.1/78.5
GAPX(cosine)	82.6/82.8	93.5/93.5	81.0/81.1	66.0/67.2	73.0/74.0	79.2/79.7
GAPX(softmax)	82.2/82.5	93.5/93.5	82.6/82.6	71.9/72.4	73.8/74.6	80.8/81.1
GAPX(maha)	84.4/84.5	92.2/92.2	81.3/81.3	66.4/68.6	72.5/73.2	79.4/80.0
GAPX(energy)	82.8/83.0	92.2/92.2	81.4/81.5	71.7/72.3	73.6/74.3	80.3/80.7

Table 8: In-distribution performance of different out-of-distribution metrics.

C Additional Discussions

C.1 Ablations on C, Eqn. 4

Model	QQP -> PIT (2.8)	PIT -> QQP (3.4)	PIT -> PAWS (32.4)	PAWS -> QQP (33.2)	PAWS -> PIT (34.5)	QQP -> PAWS (43.7)	average
GAPX(0)	62.3/63.4	72.0/72.4	55.1/55.5	70.8/70.8	69.7/70.2	52.3/54.3	63.7/64.4
GAPX(10)	63.3/64.0	72.1/72.7	48.3/51.5	69.5/69.5	69.7/70.2	47.3/52.4	61.7/63.4
GAPX(100)	66.9/67.1	72.1/72.6	48.4/50.7	71.3/71.5	71.2/71.3	52.3/54.4	63.7/64.6

Table 9: Comparing using different amount of data to set C. Macro F1/ACC are reported.

To set C in Equation. 4, in the case of 0 samples, we roughly estimate an integer from 1-5 for the interpolation constant C. For adversarial distributions like PAWS, we expect a lower perplexity because both sentences share the same bag-of-word, so we set $C = 1$. For standard distributions like QQP and WMT, we expect a modest perplexity, so we set $C = 3$. For informal distributions like PIT, where sentences do not strictly follow syntax and grammar, we expect a higher perplexity, so we set $C = 5$. In other cases, with validation data, we determine the best constant C based on the validation data (if there are multiple constants with the same results on the validation data, we take the smallest one). As shown in Table 3 (WMT results are not included here due to the lack of validation data), although using 0 or 10 samples achieve worse performances than using 100 samples, meaning the best constant threshold is not found, the results appear to be stable overall (on average 2% fluctuations in macro F1 and 1% fluctuations in accuracy). If a small amount of validation data is accessible, the performance of GAPX can be further improved.

C.2 Interpreting the Results

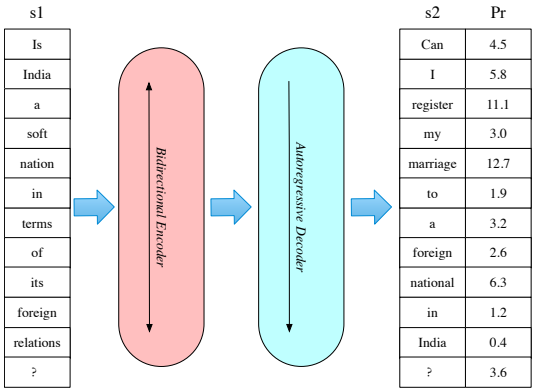

Figure 6: An example from QQP illustrating how to interpret the result of our method, by OODP.

Figure 6, Figure 7, and Figure 8 shows examples of how our autoregressive paraphrase identification models work. For OODP, our model will output a log of conditional probability for each word in s_2 given s_1 and all the previous words in s_2, namely:

$$\log P(w_2^{(i)}|s_1, Y = 1, w_2^{(1:i-1)}).$$

For IDP, we can use the log of the quotient of the conditional probability for each word given by the positive model and the negative model as an indicator which words contribute the most to the prediction result:

$$\log P(w_2^{(i)}|s_1, Y = 1, w_2^{(1:i-1)}) - \log P(w_2^{(i)}|s_1, Y = 0, w_2^{(1:i-1)}).$$

For GAP, we can use the the score defined in Eqn.. 4 split on each word, namely:

$$\log P(w_2^{(i)}|s_1, Y = 1, w_2^{(1:i-1)}) - (1 - \lambda(s_1, s_2)) \log P(w_2^{(i)}|s_1, Y = 0, w_2^{(1:i-1)}) - \lambda(s_1, s_2)C.$$

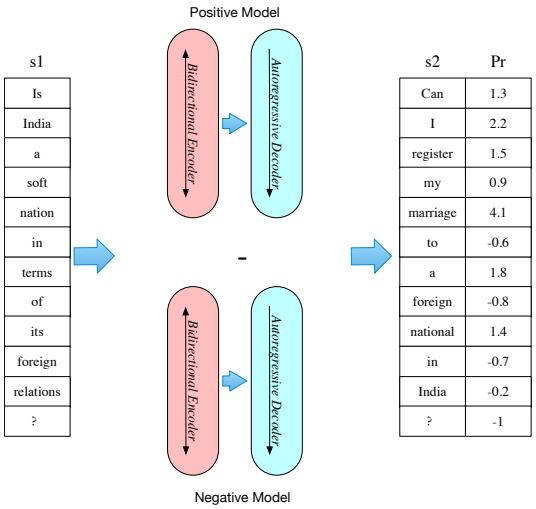

Figure 7: An example from QQP illustrating how to interpret the result of our method, by IDP.

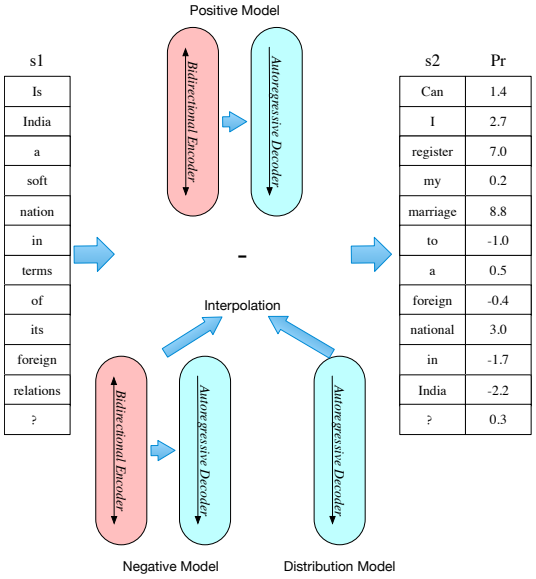

Figure 8: An example from QQP illustrating how to interpret the result of our method, by GAP.

In all three models, higher scores represent a higher chance of being non-paraphrases. For IDP and GAP, the threshold is 0 while for OODP the threshold is 3. All three models predict this sentence pair to be non-paraphrases, attending to slightly different key words. The top 3 words with the highest scores in OODP and GAP are 'register', 'marriage', and 'national'. All of them represent the words that are unlikely to occur in a paraphrase of the original sentence. The top 3 words with the highest scores in IDP are 'I', 'marriage', and 'a'. Its reliance on the word 'a' might be due to the error propagation of the autoregressive decoding.

C.3 Implementation of out-of-distribution metrics

To compare our perplexity metric with different off-the-shelf out-of-distribution metrics (Fig. 4(b) and 5(b)), we first train a RoBERTa model as described in Sec. 4.2. Both MAHA and COSINE need in-distribution validation data, for which we use half of the development data provided in each dataset. We use the other half of the development data to estimate the Weibull distribution of the

metrics. We rely on the implementation of SOFTMAX, ENERGY, MAHA, and COSINE from Zhou and Chen [65]. The metrics are calculated as follow:

1. **SOFTMAX**. We use the maximum class probability $1 - max_{j=0,1} pj$ among 2 classes (paraphrases and non-paraphrases) in the final softmax layer.
2. **ENERGY**. We use the following formula to calculate energy score:
$$g = -\log \sum_{j=0}^{1} \exp(w_j^T h),$$
where w_j is the weight of the jth class in the softmax layer, and h is the input to the softmax layer (of the concatenated (s_1, s_2) input).
3. **MAHA**. We use the input representation h of the penultimate layer of the model, and fit a Gaussian distribution to each class in the in-distribution development data $\mathcal{D}_{val} = \{(x_i, y_i)\}_{i=1}^{M}$:
$$\mu_j = \mathbf{E}_{y_i=j}[h_j]$$
$$\Sigma = \mathbf{E}[(h_i - \mu_{y_i})(h_i - \mu_{y_i})^T].$$
Then, the MAHA distance is calculated as:
$$g = -\min_{j=0,1} (h - \mu_j)\Sigma^{+}(h - \mu_j),$$
where Σ^{+} is the pseudo-inverse of Σ.
4. **COSINE**. We use the maximum cosine similarity of h (of the concatenated (s_1, s_2) input) to samples in the validation dataset:
$$g = -\max_{i=1}^{M} \cos(h, h_i^{(val)}).$$

C.4 RCA Implementation Details

To instantiate a measurement of RCA* scores. We use the RoBERTa model described in Section 4.2 as our classifier. To measure the distribution shift from a source distribution $\mathcal{D}^s$ to a target distribution $\mathcal{D}^t$. We use a training set of $\mathcal{D}^s$, a test set of $\mathcal{D}^s$, and a development set of $\mathcal{D}^t$. All the measurements of RCA* scores in this paper fix the size of the development to be 1000, and use the test set of $\mathcal{D}^s$ to be the entire test set of the original dataset.

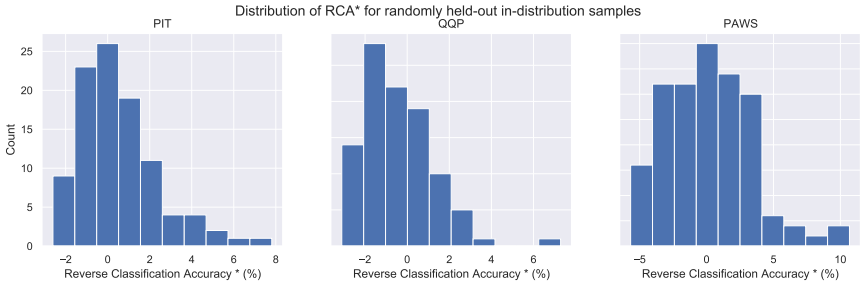

Figure 9: Distribution of RCA* for randomly held-out in-distribution samples

To calibrate the RCA* scores, we measure the distribution shift of one dataset to itself. We first train a binary classifier RoBERTa $M1$ (of paraphrases and non-paraphrase) with the training set of $\mathcal{D}^s$ for 3 epoches with an Adam Optimizer of learning rate 2e-5 (training with multiple runs with different random seeds to select the best model on a development set of $\mathcal{D}^s$, different from the development set of $\mathcal{D}^t$). Then we apply $M1$ to the development set of $\mathcal{D}^t$ to relabel those data. We take the relabeled data to retrain a classifier $M2$ and apply $M2$ to the test set of $\mathcal{D}^s$. We measure the performance drop from $M1$ to $M2$ on the test set of $\mathcal{D}^s$ as the RCA score. Note here that we make small changes to the originally proposed RCA score where the performances are measured in terms of ACC scores. We found that AUROC scores are in practice more stable for measuring RCA scores, so we use AUROC scores instead. To get the final RCA* score, we use the equation:
$$RCA*(\mathcal{D}^s, \mathcal{D}^t) = RCA(\mathcal{D}^s, \mathcal{D}^t) - RCA(\mathcal{D}^s, \mathcal{D}^s),$$

where $RCA*(\mathcal{D}^s, \mathcal{D}^t)$ is the RCA* score from $\mathcal{D}^s$ to $\mathcal{D}^t$, while $RCA(\mathcal{D}^s, \mathcal{D}^t)$ is the RCA score from $\mathcal{D}^s$ to $\mathcal{D}^t$. Under this definition, the RCA* score of one dataset to itself is defined to be 0. To repeat the measurements of distribution shift from $\mathcal{D}^s$ to itself, at each time, we hold-back 1000 random pairs from $\mathcal{D}^s$, and measure the RCA. We include plots of the distribution of the RCA* in Fig. 9 as well as the corresponding raw RCA* data below:

PIT: 4.4, 0.65, 1.9, 0.91, 5.1, 4.0, 1.1, 0.63, -0.1, -0.43, 2.9, 0.6, -1.7, 0.83, -0.27, 7.8, 1.7, 3.8, 0.78, -1.2, 2.1, 1.1, -1.3, -1.2, 1.3, -1.2, 0.032, 3.0, -2.0, -1.8, -1.2, 1.8, -2.4, 6.2, -0.47, 0.26, -0.88, -1.1, 4.2, 3.4, 3.3, 0.22, -0.25, 0.065, 2.6, 0.15, -0.93, 0.27, -0.49, -1.5, -0.79, 0.38, 2.3, 0.83, -1.4, 0.81, -0.63, -1.6, 0.97, -1.6, 0.82, 1.1, 1.9, 2.3, 0.23, -1.1, 0.72, -0.87, 1.7, 0.04, -0.38, 0.23, 1.7, -1.2, 0.1, 1.4, -0.15, 1.2, -2.6, -1.4, 0.75, 2.3, -1.2, -0.68, 5.6, -0.13, -1.1, 0.16, -0.39, 0.097, -1.9, -1.1, -0.26, -1.3, -1.3, -0.8, 0.3, 0.73, -1.9, -0.51

QQP: -0.27, 0.72, -2.3, 1.3, -2.5, 1.1, -1.1, -1.4, 2.0, -2.1, -0.77, -0.63, 0.045, -0.97, -2.1, 0.11, 1.3, -0.25, -1.1, 0.39, 0.65, 1.2, 0.87, 1.5, 0.058, 3.5, -0.039, 1.6, -1.2, 2.5, 0.64, -0.48, 2.5, -1.8, -0.52, -2.1, -0.22, -1.4, 0.97, 3.0, -2.9, -0.92, -0.42, 0.72, 1.8, -1.3, 0.63, -2.0, 0.4, -1.2, -0.65, -2.0, 0.7, 0.95, -1.3, -2.1, 1.2, -1.0, 2.5, -0.32, -1.8, -0.59, -0.016, -1.4, -1.3, 2.5, 1.4, -2.2, -1.2, 0.98, 0.93, 0.98, -2.1, -1.1, -1.8, -0.43, -0.42, 7.3, -0.75, -1.5, -0.87, -2.4, -0.61, 0.084, -1.7, -0.16, -2.4, -2.0, -1.6, -2.9, -1.7, -2.3, -1.9, -2.3, -1.7, -3.1, -1.6, 0.2, -1.7, -2.9

PAWS: -3.0, -0.0092, -0.46, -5.6, -1.7, 2.7, 0.5, 2.4, 3.6, 0.034, -4.1, -3.2, 3.6, 8.2, 0.71, -0.079, 2.9, 5.4, 1.7, -5.2, 3.2, 2.8, -3.0, 2.0, -2.2, -1.7, -0.2, 4.1, -0.22, 3.9, 1.1e+01, 1e+01, 1.2, 0.0074, -0.91, -2.2, 3.9, -2.9, -2.7, -2.8, -4.2, -4.1, -1.0, -3.1, 2.6, -0.83, 0.74, 1.7, 0.56, 2.2, 0.87, -0.5, 0.83, -5.5, 0.4, -3.0, 1.1, -3.1, 0.92, -1.6, 4.1, -3.1, -0.99, 0.79, -0.15, -1.3, 5.2, 0.18, -5.7, 3.4, 2.9, 3.2, -4.0, -2.4, -4.9, -0.51, -1.8, -2.7, -3.7, 6.6, 0.65, 2.5, 1.5, -2.3, 3.1, 1.9, 1.2, -2.1, 1.7, 0.95, 2.1, -2.7, 5.9, -4.0, 1.8, -1.8, -0.2, 4.6, -1.3, -1.0