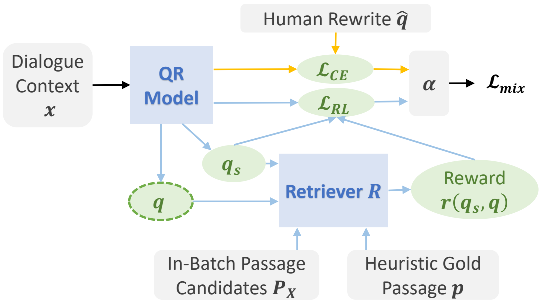

Figure 2: Our CONQRR framework. Yellow and blue arrows mark the flow of CE (**unused when** $\alpha = 1.0$) and RL loss calculation, respectively. During inference, only q (dashed border) is generated as the final rewrite.

which the QR model takes actions by generating rewritten queries and obtains rewards accordingly.

To be comparable with supervised QR models that do not use gold passages in training, we first describe how we obtain weak retrieval supervision for the RL reward calculation in CONQRR. Then we introduce the RL training details of CONQRR.

Weak Retrieval Supervision In a CQA dialogue, each question naturally comes with an answer in its following conversational utterance. For each x, we mark its weak passage label p as the one having a string span with the highest token overlap F1-score with the following answer string u_{n+1}:

$$p = \arg\max_{p' \in P} \left[\arg\max_{s \in p'} sim(s, u_{n+1}) \right] \qquad (1)$$

where s is a string span and $sim()$ calculates the token overlap score between two strings.[2] Tokens are lower-cased from the NLTK tokenizer.[3] However, as searching within all candidates in P is very time-consuming, we instead first use BM25 to retrieve the top 100 passages from P with the BM25 input being the human rewrite,[4] and then locate the best passage p from these 100 candidates.

RL Training CONQRR also has T5 as the base model architecture. It can be initialized with either T5 or T5QR. Our analysis in Section 4 shows that both setups generally work well.

For each training example with the dialogue context x, we use the concatenated utterances in x as the model input. For each input, we generate m sampled rewritten queries $(q_{s_1}, \ldots, q_{s_m})$ as well as a baseline generated rewrite q. To generate each sampled rewrite q_s, at time step t of the decoding process, a token q_s^t is drawn from the decoder probability distribution $Pr(w|x, q_s^{1:t-1})$ The baseline rewrite q is the output of greedy decoding,[5] which is also applied for query rewriting during inference. We then apply a self-critical sequence training algorithm (Rennie et al., 2017) to calculate the reward for each q_s relative to q as $r(q_s, q) = score(q_s) - score(q)$. The intuition is to reward/penalize the generation of sampled rewrites that lead to better/worse retrieval performance than greedy decoding used during inference. Ideally, the $score()$ function should be some retrieval evaluation metric like mean reciprocal rank (MRR) or Recall@K. However, as it is very costly to run actual retrieval for each training step, we instead use an approximate scoring function described below.

To compute $score(q)$ for a rewrite q, we first use q to do retrieval from the in-batch passage candidates P_X defined as follows, instead of from the full passage corpus P. We pre-compute one positive and one hard negative passage (p and p_n) for each training example x where p_n is a randomly selected passage that is different from p, 50% of the time from the top 100 BM25-retrieved candidates (with the BM25 input being the human rewrite) and remaining 50% of the time from P. We define the set of all such positive and negative passages of input examples in a batch X as the in-batch passage candidates P_X. Formally, we define $P_X = \{p^i, p_n^i | x_i \in X\}$ as the set of in-batch passage candidates for the batch X. Then for a generated rewritten query q of $x \in X$, we calculate $score(q)$ as a binary indicator of whether the retriever R ranks the assigned positive passage p highest from P_X. We denote $R(q, P_X, k)$ as the k-th most relevant passage retrieved by R from the candidate pool P_X, and define:

$$score(q) = \mathbb{1}\left[R(q, P_X, 1) = p\right] \qquad (2)$$

Then the RL training loss for x becomes:

$$\mathcal{L}_{RL} = -\frac{1}{m} \sum_{i=1}^{m} r(q_{s_i}, q) \log Pr(q_{s_i}|x)$$

$$Pr(q_{s_i}|x) = \prod_{t=1}^{|q_{s_i}|} Pr(q_{s_i}^t | x, q_{s_i}^{1:t-1})$$

[2] We randomly choose a passage if there is a tie in scores.

[3] https://www.nltk.org

[4] We show in Section 4.3 that using the dialogue context as the BM25 input to induce weak supervision gives similar performance (Figure 3), where no human rewrites are used.

[5] We tried beam search with various beam sizes and got similar results as greedy decoding.

Following prior work (Paulus et al., 2018; Celikyilmaz et al., 2018), we experiment with a pure RL loss ($\mathcal{L}_{RL}$) and a mixed RL and CE loss in training:

$$\mathcal{L}_{mix} = \alpha\mathcal{L}_{RL} + (1-\alpha)\mathcal{L}_{CE} \qquad (3)$$

where $\alpha \in [0, 1]$ is a tunable parameter.

Inference At inference time, both T5QR and ConQRR work in the same way. The trained QR model greedily generates the rewritten query given a dialogue context. Then, the predicted rewrite is given to the provided retriever to perform retrieval.

3.3 Retriever Models

We evaluate the effectiveness of ConQRR in experiments with two general-domain retrieval systems, with more details in Appendix A.1.

BM25 We follow Anantha et al. (2021) using Pyserini (Yang et al., 2017) with default parameters $k1 = 0.82$ and $b = 0.68$.

Dual Encoder (DE) We use a recent T5-base dual encoder model (Ni et al., 2021) which achieves state-of-the-art performance on multiple retrieval benchmarks. This model is fine-tuned on MS MARCO, and kept fixed for our experiments.

4 Experiment

Dataset QReCC (Anantha et al., 2021) is a dataset of 14k open-domain English conversations in the format of alternating user questions and agent-provided answers with 80k question and answer pairs in total. The conversations are collected from different sources: QuAC (Choi et al., 2018), Natural Questions (Kwiatkowski et al., 2019) and TREC CAsT-19 (Dalton et al., 2020) with additional annotations by crowd workers. See more details and statistics in Appendix A.2. Therefore, QReCC can be divided into three subsets for evaluation. We name them as *QuAC-Conv*, *NQ-Conv* and *TREC-Conv* respectively to differentiate them from the original datasets from which they are derived. TREC-Conv only appears in the test set. Each user question comes with a human-rewritten query. For each agent turn, gold passage labels are provided if any. The entire text corpus for retrieval contains 54M passages, segmented in the released data.[6]

	Original Eval			**Updated Eval**		
QR Model	MRR	R10	R100	MRR	R10	R100
GPT2 + WS	0.152	24.7	41.5	0.304	49.6	83.1
Transformer++	0.155	24.8	40.6	0.311	49.8	81.4
T5QR	0.164	26.2	42.3	0.328	52.5	84.7
ConQRR (mix)	0.186	29.2	**45.0**	0.373	58.5	**90.2**
ConQRR (RL)	**0.191**	**30.0**	44.4	**0.383**	**60.1**	88.9
Human	0.199	32.8	49.4	0.398	62.6	98.5

Table 2: Passage retrieval performance of QR models, comparable to scores in Anantha et al. (2021) by using the same BM25 retriever for QReCC test set. ConQRR achieves *state-of-the-art* results. Recall@10 and Recall@100 are abbreviated as R10 and R100.

Evaluation Metrics Following (Anantha et al., 2021), we use mean reciprocal rank (MRR), Recall@10 and Recall@100 to evaluate the retrieval performance by using the provided evaluation scripts.[7] We use their *updated* evaluation script for most experiments, except that we also use the *original* version for calculating scores in Table 2 to compare with their reported QReCC baseline results. We note that these two evaluation scripts only differ by a scaling factor[8] so they should lead to the same conclusions regarding model comparisons. See more details in Appendix A.3.

Implementation Details Following prior work on RL for text generation (Paulus et al., 2018; Fisch et al., 2020), we first initialize ConQRR with a supervised model (T5QR) (Lin et al., 2020) as a warm-up. Our RL optimization (self-critical sequence training (Rennie et al., 2017)) uses a policy gradient method with Monte Carlo sampling. In Section 4.3, we show that although initializing with T5QR works better than T5, both setups generally work well. All our models use T5-base as the base model. We experiment with ConQRR trained with either a mixed ($\mathcal{L}_{mix}$) or pure RL ($\mathcal{L}_{RL}$) loss. For the mixed loss, we observe that ConQRR works well when the RL loss weight α is large.[9] We tune its values in 0.9, 0.95, 0.97, 0.99, and use 0.99 as the final value. Due to space limit, more implementation and hyper-parameter details are reported in Appendix A.1.

[6] Original QReCC data: `https://zenodo.org/record/5115890#.YZ8kab3MI-Q`.

[7] Both original and updated evaluation scripts: `https://github.com/scai-conf/SCAI-QReCC-21`.

[8] This is due to the exclusion of test examples with no valid gold passage labels (roughly 50%) in the updated evaluation, which results in 6396, 1442 and 371 test instances for QuAC-Conv, NQ-Conv and TREC-Conv, respectively.

[9] We also experiment with $\alpha = 0.0$, where the RL loss is removed for both retrievers, and get similar results as T5QR.

		QReCC (Overall)			QuAC-Conv			NQ-Conv			TREC-Conv (OOD)*		
QR Model	**IR System**	MRR	R10	R100	MRR	R10	R100	MRR	R10	R100	MRR	R10	R100
T5QR	BM25	0.328	52.5	84.7	0.33	52.7	85.0	0.345	54.2	83.9	**0.230**	44.5	82.3
ConQRR (mix)	BM25	0.373	58.5	**90.2**	0.379	59.2	**90.9**	**0.385**	**58.8**	**88.9**	0.229	**44.7**	**82.7**
ConQRR (RL)	BM25	**0.383**	**60.1**	88.9	**0.395**	**61.6**	90.2	0.378	58.0	86.7	0.198	43.5	75.9
Human Rewrite	BM25	0.398	62.6	98.5	0.403	62.9	98.4	0.408	63.8	99.0	0.273	53.8	98.9
T5QR	DE	0.361	56.2	75.9	0.349	55.7	76.1	0.417	58.7	74.2	0.343	55.9	79.2
ConQRR (mix)	DE	0.395	61.9	81.8	0.387	62.0	82.4	0.439	62.2	79.0	**0.361**	**58.9**	**81.0**
ConQRR (RL)	DE	**0.418**	**65.1**	**84.7**	**0.416**	**65.9**	**85.8**	**0.453**	**64.1**	**80.9**	0.327	55.2	79.6
Human Rewrite	DE	0.422	64.8	84.0	0.409	64.5	84.1	0.483	65.8	83.2	0.411	66.0	86.5

Table 3: Passage retrieval performance on QReCC test set and 3 subsets. ConQRR (mix) beats the supervised T5QR model on all retriever system and test set combinations. * OOD (out-of-domain): only appear in the test set.

4.1 Compared Systems

For QR models, we compare three supervised models including **GPT2 with weak supervision (WS)** (Yu et al., 2020), a GPT2-medium based system that additionally leverages search sessions to create weak supervision for QR training before fine-tuning, **T5QR** (Lin et al., 2020) and **Transformer++**, the previous state-of-the-art model based on GPT2-medium (Vakulenko et al., 2021) and reported in the original dataset paper (Anantha et al., 2021), as well as **ConQRR (mix/RL)** with a mixed ($\mathcal{L}_{mix}$) or pure RL ($\mathcal{L}_{RL}$) loss. For analysis purposes, we also report performance for directly using the concatenated dialogue context as the retriever input without any query rewriting in Section 4.3. We experiment with two off-the-shelf retrievers, **BM25** and **DE** (Section 3.3).

4.2 Quantitative Results

To have a direct comparison with the original QR baseline Transformer++, which has the retrieval performance reported on the overall QReCC test set by using BM25 as the off-the-shelf retriever, we first compare all QR models in the same setting in Table 2 and use both the original and updated versions of the provided evaluation script. GPT2 + WS has similar performance as Transformer++. T5QR and ConQRR outperform the Transformer++ baseline by 5% and 18% respectively, averaged on three metrics,[10] although Transformer++ is based on a larger base model - GPT2-medium. Therefore, ConQRR (RL) becomes the *state-of-the-art* QR model for conversational passage retrieval on QReCC with the original BM25 retriever in Anantha et al. (2021).

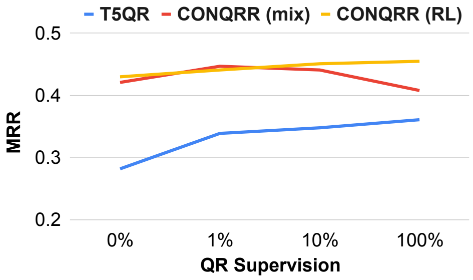

Figure 3: MRR on QReCC versus the percentage of QR supervision used for training, with DE as the retriever.

Table 3 shows more comprehensive retrieval results comparing ConQRR and the supervised model T5QR, with the updated evaluation script. For the overall QReCC test set, ConQRR outperforms T5QR for all three metrics. For MRR and Recall@10, gains are roughly 15% with the RL loss and 9-14% with the mixed loss for both retrievers. Gains in Recall@100 vary more (4-12%). Breaking down the results by subset shows that the mixed loss is more robust. ConQRR (RL) is less effective for the TREC-Conv subset, which only appears in the test set. This suggests that RL loss alone does not generalize well to out-of-domain examples. Across all subsets, the best MRR and Recall@10 results are consistently from DE, whereas BM25 has better Recall@100 scores. See our explanation in Appendix A.4.

4.3 Analysis

Zero or Few QR Supervision We investigate how sensitive ConQRR and T5QR are to the availability of QR labels. We experiment with training T5QR with 0%, 1%, 10% or 100% of QR labels in the QReCC train set. For the case of 0% examples, we simply use the original T5 checkpoint without fine-tuning. When training ConQRR, we mask

[10] We obtained prediction results from the authors and reran their evaluation script. The numbers we got are slightly lower than what they reported, but do not affect the conclusions.