BERT-based Passage Retrieval and Ranking on Texts with Typos

Quincy Hughes
qhughes22@amherst.edu

Jung Hoon Lim
junghoonlim@umass.edu

Dennis Minn
dminn@umass.edu

Christopher Shi
cshi@umass.edu

1 Problem statement

In recent years, NLP models utilizing the transformer architecture have shown great performance for various NLP tasks on benchmark datasets. While this has been the case in objective research settings, in industrial NLP applications it is impossible to guarantee the absence of noise in datasets. As such, the task of evaluating the performance of NLP models and their effectiveness on common NLP tasks in the presence of noise is crucial in understanding and fine-tuning NLP models. In practice, it has been shown that for most NLP models based on a transformer architecture, as the level of noise increases in text data sets the performance degrades as well (Bagla et al., 2021).

The extent to which information retrieval systems lose effectiveness in the face of noise is not clear. In the most common application of information retrieval systems, the search engine, users input a query and a list of webpages relevant to that query is returned. With both queries and documents in this context being prone to typographical errors, the effect of such errors on model effectiveness is an important question.

We aim to analyze the effects of noise on performance by three BERT-based passage retrieval models: ColBERT (Khattab and Zaharia, 2020), Dense Passage Retrieval (Karpukhin et al., 2020), and Reranker (Nogueira and Cho, 2019). In particular, we study how resilient these models are to typographical errors in both queries and passages.

2 What you proposed vs. what you accomplished

Since our original proposal, the following changes were made based on the feedback provided for our proposal.

- We decided not to do error corrections and concentrate on the information retrieval part using MS Marco dataset as there would be two objectives if we also correct the typos
- We increased the complexity of noise, expanding beyond character level noise
- We used recall@k as an additional evaluation metrics

Considering the changes mentioned above, we proposed the following things. The things that we accomplished are marked with strikethrough.

- ~~Collect and preprocess the MS Marco dataset~~
- ~~Build and train Reranker, Dense Passage Retriever, and ColBERT~~
- ~~Use the three models for passage retrieval and passage reranking task~~
- Make the evaluation results for MRR@k and Recall@k of our baseline model similar to results in papers: We wrote the details about this problem in section 6.7
- ~~Generate noisy texts and evaluate the three models' performances the noisy texts~~
- ~~Perform in-depth error analysis to figure out what kinds of examples our approach struggles with~~

3 Background

3.1 Information Retrieval

Sparse Representations Also referred to as lexical representation, spare representations quantify sentences into vectors by representing each word as its own dimension/column. Since the dimensionality of the vector is comparable to the vocab-

ulary of the training set, the vector is comprised of mostly zeros, hence the name.

Dense Representations Dense representations are vectors represented by projecting sentences into a dimension lower than the vocabulary. A good projection should encode sentence semantics and syntax with two semantically related sentences close together in the project vector space (Mikolov et al., 2013).

3.2 Models

Many information retrieval models that give the best performance use dense representations of each text to calculate the relevance between queries and passages using variants of the Transformer(paper) model. Some examples of similarity measurements between vectors can be dot products, cosine similarity, L2 distance, and Mahalanobis distance.

Cross Encoder This paradigm re-purposes BERT as a passage re-ranker and generally follows the sentence entailment schema presented in the original BERT paper. The query and passage texts are analogous to sentence A and sentence B respectively. By formatting the input as the concatenation of the query and passage, the self-attention mechanism effectively mirrors the bi-directional cross attention mechanism seen in (Vaswani et al., 2017) between the two texts, hence the name cross encoder (Devlin et al., 2018).

Bi-Encoder A variation of the siamese network architecture that replaces the feed-forward neural networks with a pretrained BERT model. The core concept of the siamese architecture is to apply the same model on two different inputs in tandem and have their respective outputs are compared using a similarity function (Euclidean distance, cosine similarity, etc.). For bi-encoders in the question-answering domain, the underlying BERT model separately encodes the query and passage, and the query's relevancy with respect to the passage is quantified using a similarity function. The produced score can be considered the semantic similarity between the projected representation of the two input vectors (Chicco, 2021).

While trying various kinds of passages as negative instances for training dual-encoder models, (Karpukhin et al., 2020) discovered that using in-batch negatives for training gives a significant performance gain compared to previous approaches. In-batch training refers to using passages that are relevant to other queries as irrelevant passages inside the same batch while training. Using in-batch negatives also give computational efficiency as we can use each of the encoded passage vectors B times to calculate B losses when the batch size is B.

(Karpukhin et al., 2020) further discovered that adding "hard negatives", which are passages that has high BM25 score to a query but is not relevant to the query, to the in-batch negative setting gave another significant performance boost.

ColBERT (Khattab and Zaharia, 2020) introduce a "best of both worlds" option named ColBERT. ColBERT uses late interaction to delay the query-document interaction, allowing for cheaper passage re-ranking and end-to-end neural retrieval. ColBERT's architecture consists of a query encoder, a document encoder, and the late interaction mechanism. The query and document encoders share a BERT model.

This approach allows passage embeddings to be computed offline, and it allows for pruning using fast vector-similarity algorithms to skip passages without considering the complete interaction matrix. ColBERT achieves similar effectiveness at passage re-ranking compared to BERT-based models, but does so with 170× lower latency and 13,900× lower FLOPs. It can also be used for end to end passage retrieval, significantly outperforming the bag-of-words BM25 model, though at higher latency.

4 Related Work

The goal of information retrieval is to define a function that identifies semantically similar text. Prior to BERT, the most models used sparse representations (e.g. TF-IDF, bag-of-word) (Zhang et al., 2010) with most search engines employing variations of BM-25 (Robertson and Zaragoza, 2009). The main issue with sparse representation is that it failed to address lexical gaps (e.g. running, ran, run), word ambiguity (e.g. river bank or financial bank), or word order.

(Mikolov et al., 2013) did address these problems, however dense representations in information retrieval were not widely utilized until the publication of BERT (Devlin et al., 2018). (Nogueira and Cho, 2019) used BERT as a passage reranker re-purposing the [CLS] BERT embeddings to quantify relevancy. While this model has been shown to significantly outperform BM-

25, its main drawback is its long computation time (Thakur et al., 2021).

(Reimers and Gurevych, 2019) remedies this issue by presenting the bi-encoder architecture with cached dense representations. By pre-computing and indexing the passages' BERT embeddings, (Reimers and Gurevych, 2019) can rank 10,000 sentences in 5 seconds where it would take (Nogueira and Cho, 2019) approximately 65 hours. The bi-encoder architecture employing BERT also outperform BM-25 (Thakur et al., 2021), but remains outperformed by the cross encoder.

(Qu et al., 2020) optimizes the bi-encoder architecture by introducing hard-negatives, converting the training task from binary classification to multi-label classification, and then applying the margin-ranking loss function between the bi-encoder and cross-encoder predicted relevancy scores. (Karpukhin et al., 2020) simplifies this approach omitting the margin-ranking loss step while achieving comparable performance.

(Khattab and Zaharia, 2020) also attempted to minimize the performance gap between the bi-encoder and cross-encoder by modifying tokenization schema. However, the main contribution of (Khattab and Zaharia, 2020) was the use of the late-interaction mechanism in favor of previous score functions (e.g. dot product, cosine similarity, etc.).

In our paper we benchmark these leading models presented by (Nogueira and Cho, 2019), (Karpukhin et al., 2020), (Khattab and Zaharia, 2020) and then re-evaluate performance in the presence of noise. Our approach follows (Zhuang and Zuccon, 2021) and evaluate the performance on noisy texts, but expands the paper by testing on multiple models architectures and by increasing the complexity of noise to beyond character level noise.

5 Dataset

We used the MS Marco dataset introduced in (Nguyen et al., 2016). The MS MARCO queries consists of questions pulled from the Bing search engine. The MS MARCO passages came from web documents which provide the information necessary to answer the questions. The MS Marco dataset is structured in such a way that the training data has about one or few relevant training passage per training query.

For passage ranking, passages that contains answer the desired query are manually selected by editors. Given a query, our task involves ranking the relevancy of a set passages such that the human selected passage is identified in our predicted k most relevant passages.

The main challenge of our dataset is filtering out passages that they are somewhat related to query, but does not answer the question. (**Table 1**) and (**Table 2**) demonstrates multiple passages that talk about the same topic, but ultimately there is only 1 relevant passage for query with id 1923.

Queries and Passages Queries and Passages are given in the format of (text id, text). There were in total of 1,010,916 queries that are split into training, dev, and eval set with approximately 8:1:1 ratio. For passages, there were 8,841,823 passages. However, we didn't use the eval set because the relevant passages for the eval set was not open to public in the official website.

pid	passage
0	The presence of communication amid scientific minds was equally important to the success of the Manhattan Project as scientific intellect was...
1	The Manhattan Project and its atomic bomb helped bring an end to World War II. Its legacy...
2	The Manhattan Project was the name for a project conducted during World War II, to develop the first atomic bomb. It refers specifically...

Table 1: Passage

qid	query
121352	define extreme
634306	what does chattel mean on credit history
19232	**answer Identify the scientist who became the director of the Manhattan Project, which created the atomic bomb**

Table 2: Queries

Triples The format of this dataset is a query, a relevant passage, and an irrelevant passage text.

The full triples file (triples.train.full.tsv) and small triples file (triples.train.small.tsv) has about 400 million and 40 million triples respectively. For our final dataset, we used the qidpidtriples file (qidpidtriples.train.full.tsv), which has about 270 million triples. The format of the qidpidtriples file is slightly different in that text ids are used instead of texts. We used the small triples file and the qidpidtriples file for training and validation. An example of triples files are shown in (Table 3).

query	positive passage	negative passage
is a little caffeine ok during pregnancy	... If you're pregnant, limit caffeine to 200 milligrams each day...	It is generally safe for pregnant women to eat chocolate...

Table 3: Triples

Top1000 The format of this dataset is query id, passage id, query text, passage text.

This dataset shows top 1000 passages retrieved by using BM25 for each query. The format of this dataset is query id, passage id, query text, passage text. However, many queries don't have exactly 1000 retrieved passages and this makes sense as there could be less than 1000 passages that have BM25 scores of 0. An example of Top1000 file is shown in (**Table 4**).

qid	pid	query	passage
188714	1000052	foods and supplements to lower blood sugar	Watch portion sizes: Even healthy foods will cause high blood sugar...

Table 4: Top1000

Qrels The name is an abbreviation for query relevancy. The format[1] of this dataset is query id, iteration number, passage id, relevancy. The iteration number is ignored. The relevancy value always 1, dictating that the given passage corresponds with the given query. An example of Qrels file is shown in (**Table 5**).

qid	Iter #	pid	Relevancy
188714	0	2026790	1

Table 5: Qrels

5.1 Data preprocessing

We made our final training and validation data using the qidpidtriples file. For training data, we excluded all cases where the negative passage in each triple is relevant to any other query in the training set. This was done to avoid cases where there could be two relevant passages for one query during the in-batch training. For reranker, each triple gave us one relevant and one negative (query, passage) pairs.

[1]The official MS MARCO passage ranking dataset states "TREC qrels format", but didn't state which year. Thus, we followed the 2014 TREC format because this was only available description that we could find.

The details about noise generations are written in **section 7.1**.

5.2 Data annotation

Our project did not involve data annotation.

6 Baselines

Our baseline requires that we build each model architecture from scratch. In order to accurately compare the performance of each model, we believed that each model should have a comparable number of parameters. Unfortunately pretrained ColBERT[2] used distilBERT as its backbone, DPR used the base BERT model, and Reranker used the large BERT model.

The alternative benchmarking resource (Thakur et al., 2021) does present the above models in a fair and comparable setting, but only for zero-shot learning. Employing (Thakur et al., 2021) would leave the cause of performance degradation inconclusive: was the performance degradation caused by the fact that these models had never seen MS MARCO dataset or was it cause by the data corruption?

For our baseline, we evaluated the performance (**Table 6**) of each model architecture mentioned in **section 3.2**, using the base BERT model as the backbone, on the original passage ranking MS MARCO datasets (Nguyen et al., 2016). Shown in (Thakur et al., 2021),the top performing passage ranking models utilize dense representations, thus we chose to use these architectures over models that use sparse representations.

For training, we used learning rate of 3e-7 for all models and used the same linear learning rate scheduling scheme where we warm up for 20,000 steps and set the learning rate to be linearly decayed to 0 at 200,000 steps. We used the checkpoints that gave lowest loss as our trained baseline models. We used lower batch size due to the GPU memory constraint that we have. The GPU that we used had 24 GB of memory.

We used a WordPiece tokenizer provided by the HuggingFace package that was used for the pretraining. For optimizer, we used the Adam optimizer ((Kingma and Ba, 2014)) with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and L2 learning rate decay of 0.01. **Section 7.3** goes over the details regarding the specific libraries.

[2]Hugging Pretrained Models

	Reranker			DPR		ColBERT		
Noise-Type	MRR@10	MRR@1000	Recall@10	MRR@10	Recall@1000	MRR@10	MRR@1000	Recall@10
Original	**0.2891**	**0.2990**	**0.6562**	**0.2460**	**0.9039**	**0.2798**	**0.2903**	**0.6386**
[Q] Backtranslation	0.2591	0.2696	0.6002	0.2115	0.8539	0.2498	0.2608	0.5856
[Q] Lemmatize(0.5)	0.2860	0.2962	0.6480	0.2368	0.8909	0.2747	0.2854	0.6284
[Q] Lemmatize(1.0)	0.2829	0.2932	0.6417	0.2270	0.8771	0.2701	0.2809	0.6182
[Q] NeighCharSwap(one)	0.1963	0.2082	0.4805	0.0932	0.5759	0.1833	0.1956	0.4521
[Q] RemoveSpace(one)	0.2425	0.2534	0.5692	0.1781	0.7796	0.2335	0.2449	0.5484
[Q] RemoveStop(one)	0.2842	0.2942	0.6518	0.2330	0.8949	0.2728	0.2834	0.6273
[Q] RemoveStop(0.5)	0.2833	0.2935	0.6469	0.2327	0.8921	0.2706	0.2812	0.6257
[Q] WordOrderSwap(ones)	0.2853	0.2953	0.6483	0.2285	0.8911	0.2671	0.2780	0.6179
[Q] WordOrderSwapAdj(ones)	0.2868	0.2970	0.6497	0.2339	0.8983	0.2714	0.2821	0.6274
[P] Backtranslation	0.2850	0.2952	0.6432	0.2404	0.9008	0.2708	0.2814	0.6248
[P] Lemmatize(0.5)	0.2877	0.2977	0.6518	0.2430	0.9025	0.2786	0.2892	0.6308
[P] Lemmatize(1.0)	0.2841	0.2941	0.6501	0.2382	0.9011	0.2739	0.2846	0.6268
[P] NeighCharSwap(0.1)	0.2646	0.2750	0.6147	0.2031	0.8479	0.2619	0.2724	0.5929
[P] RemoveSpace(0.1)	0.2739	0.2843	0.6226	0.2234	0.8769	0.2632	0.2739	0.6021
[P] RemoveStop(0.2)	0.2874	0.2976	0.6492	0.2413	0.9004	0.2730	0.2839	0.6244
[P] RemoveStop(0.5)	0.2761	0.2863	0.6406	0.2272	0.8918	0.2552	0.2664	0.6001
[P] WordOrderSwap(ones)	0.2788	0.2893	0.6383	0.2338	0.8910	0.2694	0.2802	0.6165
[P] WordOrderSwapAdj(ones)	0.2884	0.2985	0.6521	0.2410	0.9020	0.2754	0.2859	0.6304

Table 6: Evaluation Results

Please refer to **section 7.2** for the abbreviations. [Q] indicates that the query text was corrupted. [P] indicates that the passage text was corrupted

6.1 Reranker

To evaluate the cross-encoder model, we decided to use the model presented in (Nogueira and Cho, 2019) named Reranker.

The input for reranker was tokenized in the form of ([CLS], query tokens, [SEP], passage tokens, [SEP]). The [CLS] vector passed through a linear layer to get scores for a classification task with two classes. For input type ids, we used 0 until the first [SEP] token and 1 for the rest of the tokens. This scheme is identical to the next sentence prediction in the original of (Devlin et al., 2018) except that we use queries and passages instead of sentences. In Reranker, the two classes are whether the input query and passage were relevant or irrelevant.

In (Nogueira and Cho, 2019), the query tokens were truncated to have a maximum length of 64 and the overall length of tokens were truncated to have a maximum length of 512. However, because not truncating query tokens to have a maximum length of 64 gave us slightly better results in the training and validation accuracies, so we didn't truncate the query tokens for our baseline model.

The loss can be written as

$$L = -\sum_{j \in J_{\text{NEG}}} \log(s_j) - \sum_{j \in J_{\text{POS}}} \log(1 - s_j)$$

where J_{NEG} and J_{POS} are set of indices of irrelevant and relevant passages.

We used a batch size of 24 and the best result was given at the checkpoint saved at 116,000 steps.

6.2 Dense Passage Retrieval

To evaluate the bi-encoder model, we decided to use the model presented in (Karpukhin et al., 2020) named Dense Passage Retriever (DPR). In particular, (Karpukhin et al., 2020) used one d-dimensional real-valued dense vector to represent each text. While the representation of the [CLS] token or mean/max pooling across all contextual output representations can be used, (Karpukhin et al., 2020) uses the [CLS] token as shown in their open-source codes. (Karpukhin et al., 2020) showed that in their implementation, while L2 performs comparable to dot product and gave better results than cosine similarity. Also, it showed that using negative log likelihood based on softmax gives slightly better retrieval accuracy than using triplet loss. Following the results shown by (Karpukhin et al., 2020), we used the [CLS] token for the dense vector and dot product for the relevance score. We used a query encoder and a passage encoder to encode queries and passages respectively. The input texts are tokenized in the shape of ([CLS], text tokens, [SEP]) where either a query or a passage is used for the text.

For training, we used in-batch training with additional hard negatives. We used the positive passages in the triples files for in-batch training and the negative passages as the additional hard negatives. The negative passage in triples are selected from the top 1000 passages based on BM25. When the batch size is B, each query has to choose

the the relevant passage among $2B$ passages. We used the cross-entropy loss of this classification problem for training. The loss function can be written as

$$L(q_i, p_i^+, p_{i,1}^-, \ldots, p_{i,2B-1}^-)$$
$$= -\log \frac{e^{\text{rel}(q_i, p_i^+)}}{e^{\text{rel}(q_i, p_i^+)} + \sum_{j=1}^{2B-1} e^{\text{rel}(q_i, p_{i,j}^-)}}$$

where p_i^+ and $p_{i,j}^-$ refer to the positive and jth negative passage for query q_i and $\text{rel}(q_i, p_i)$ is the relevance score between query q_i and passage p_i.

We used a batch size of 22 and we used the checkpoint saved at 68,000 steps as our baseline model.

6.3 ColBERT

While the Dense Passage Retrieval model use two encoders, ColBERT uses one encoder and let the model know whether the input text is query or a passage by adding another token [Q] or [D] for queries and passages respectively. Also, the length of the query tokens are set to N_q by adding [MASK] tokens if the query is short and truncating the query tokens if the query is long. The resulting form of the input tokens for queries is ([CLS], [Q], q_0, q_1, ..., q_l, [MASK], [MASK], ..., [MASK], [SEP]). The input for passages has the form of ([CLS], [D], p_0, p_1, ..., p_n, [SEP]). Following the number that is used in the original paper, we used 32 for N_q. The output of the BERT base model gives a vector with size of 768 for each input token. These vectors are passed through a linear layer to produce a 128 size vector. The vectors with size of 128 will be used to calculate the relevance score between queries and passages.

The relevance score between each query and passage is determined by the late interaction algorithm proposed by the paper. First, for passages, the encoded vectors of the punctuation characters are removed. Second, the relevance score for each query token is calculated by choosing the maximum cosine similarity between the encoded vector of the query token and each remaining passage vectors. Finally, the sum of the relevance score for each query token is used as the relevance score between each query and passage.

For training data, we used the same in-batch training with additional hard negatives as in the DPR. Accordingly, loss function is the same except for how the similarity scores are calculated.

We used a checkpoint saved in 176,000 training steps that was trained using a batch size of 24.

6.4 Training, validation, test split

We used the dev dataset of MS Marco provided by the official website to evaluate our models. For training, we split the official train dataset into training and validation datasets based on queries and the dev dataset was used for the evaluation. In total, there were 808,731 queries in the official train dataset and we randomly selected 48,000 to use exclusively for the validation. The validation was done every 1,000 steps using a part of the validation data with at least 22,000 instances, which was equivalent to 1000 batches. The accuracy and loss was measured and checkpoints with the 10 lowest validation losses were saved. After choosing the best based on the validation loss, we performed evaluation using the offical dev set by measuring MRR and Recall for reranking or retrieval task.

6.5 Evaluation Metrics

We used mean reciprocal rank at k (MRR@k) and recall at k (Recall@k) to evaluate all models.

The mean reciprocal rank is the averaged reciprocal minimum rank of relevant passages for a set of given queries. The reciprocal rank is calculated using the multiplicative inverse of the predicted rank of the highest ranked target passage.

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$$

Q in the above equation is the number of queries. MRR@k is the MRR when only the top k passages are proposed, we use 0 instead of the reciprocal rank if the minimum rank is bigger than k.

Recall@k is the proportion of relevant items found in the predicted top k results.

$$\text{Recall@k} = \frac{\#(\text{relevant items retrieved in top } k)}{\#(\text{relevant items})}$$

We calculated the average of recall@k for each query where queries with no relevant passages are ignored in the calculation.

For Reranker and ColBERT, we measured MRR@10 and Recall@10 for the reranking task, where we rerank the top 1000 retrieved passages using BM25. For DPR, we measured MRR@10

and Recall@1000 for the retrieval task where the we retrieve the top 1000 passages from 8.8 million passages for each query.

6.6 Baseline results

The baseline results for the three models on unmodified texts are mostly shown in (**Table 6**) in the row that starts with 'Original'. For the reranker model, MRR@10 and Recall@10 for the reranking task was 0.2891 and 0.6562 respectively. For the DPR model, MRR@10 and Recall@1000 for the retrieval task was about 0.2460 and 0.9038 respectively. While not shown in (**Table6**) MRR@10 and Recall@10 for the DPR model for the reranking task was about 0.2559 and 0.5959 respectively. Finally, for the ColBERT model, MRR@10 and Recall@10 for the reranking task was about 0.2798 and 0.6386 respectively.

6.7 Performance

The performance of our baseline models are shown in **Table 6**. However, the performance was about 15-20% lower MRR@10 compared to figures that can be found in papers. The reason for this may be because of the lower batch size that we used or less training data that are used. Regardless, to figure out the reason, we tried various configuration for the training.

First, we tried different learning rates and learning rate scheduling. As a result, we decided to use a learning rate of 3e-7 for all three models. Lower learning rate could give slightly better results, but we restricted the maximum training steps to 200,000 due to the limited computation source. The learning rate scheduling didn't make significant effect on performance.

Second, we tried different models and packagees. For the reranker, instead of making the model from scratch, we tried the BertForNextSentencePrediction model from Huggingface, which is a model that is already built for the next sentence prediction task which uses identical model architecture as our reranker model. For Dense Passage Retrieval, we tried by setting the dense vector for each text to be the average of the output vectors from BERT. We also re-wrote our codes without the Pytorch-Lightning package that our initial codes used. The results didn't change significantly in any cases.

Third, we tried on different type of datasets. We used the train small file and qidpidtriples file with and without shuffling data, but the results didn't change. For in-batch training, we made our own dataset with more difficult "hard negatives" by reranking the top 1000 BM25 results using BM25 to get more exact rankings of each passage to a query. Then, we used only the top 20 results for each query after excluding every passages that are relevant to any query in the training set. However, the accuracy of the training dropped significantly and the resulting MRR@10 also dropped significantly. We concluded that giving too hard samples for hard negatives may result in worse performance.

Fourth, when parsing through some of text, we noticed a number of weird symbols appearing in sentences with conjunctions[3]. As an example, it seemed like all apostrophes were not being recognized as usual. Eventually, we cleaned many characters that were recognized as more than one byte in utf-8 such as Latin-1 characters to a similar special character. However, this didn't gave us better performance as the training curve was almost identical as before. We concluded that either the characters such as apostrophes were not important in dense vectors or the tokenizer already has handled these characters.

7 Approach

7.1 Synthetic Typo Generation

As a continuation of the work done by Zhuang and Zuccon (2021), we sought to introduce noise not only at the character level, as already done by Zhuang and Zuccon (2021), but also introduce noise at the word and grammatical level. The techniques utilized within our synthetic noise generation follow two different dogmas: noise generation techniques that seek to introduce character level noise, and noise generation techniques that seek to introduce word/grammatical level noise. The following typo/noise generation techniques seek to introduce character level noise:

- **Neighboring Character Swap (NeighCharSwap):** Randomly swaps a character with one of its neighbor characters, e.g., "search the typos" → "search the tyops".
- **Remove Spaces (RemoveSpace):** Randomly removes the space between two words, e.g., "search the typos" → "searchthe typos"

[3]We followed the solution presented in github issue 945 due to its similiarity to our issue. However this solution only addressed some of the incorrectly decoded character.

The following typo/noise generation techniques seek to introduce word or grammatical level noise:

- **Word Order Swap (WordOrderSwap):** Randomly swaps the location of one word with the location of another word, e.g., "search the typos" → "typos the search".
- **Word Order Swap Adjacent (WordOrderSwapAdj):** Randomly swaps the location of one word with one of its neighboring words, e.g., "search the typos" → "search typos the".
- **Lemmatize (lemmatize):** Randomly lemmatizes a word utilizing the NLTK word net lemmatizer (Bird et al., 2009), e.g., "search the typos" → "search the typo".
- **Remove Stopwords (RemoveStop):** Randomly removes stop words utilizing the NLTK stop words corpus (Bird et al., 2009), e.g., "search the typos" → "search typos".
- **Backtranslation (Backtranslation):** Using French as the pivot language, backtranslates the input text utilizing the googletrans package, e.g., "search the typos" → "check for typos".

Since queries in MS MARCO are relatively short (roughly 6 keywords on average) (Nguyen et al., 2016), when generating typos for queries, we only considered keywords that have more than 3 characters or words with the maximum number of characters if there are no word longer than 3. Also, each sentence is made by splitting texts by the character '.'. The types of noises that we generated for queries and passages are written below where $p \in [0, 1]$, 'one', 'ones' refer to introducing noise by randomly modifying each word with a predefined probability p, modifying one word per sentence, and modifying one word only. For backtranslation, the each text was backtranslated as a whole.

- **Backtranslation:** (each text), (each text)
- **Lemmaztize:** (0.5, 1), (0.5, 1)
- **Neighboring Character Swap:** (0.1), (one)
- **Remove Spaces:** (0.1), (one)
- **Remove Stopwords:** (0.2, 0.5), (one, 0.5)
- **Word Order Swap:** (ones), (ones)
- **Word Order Swap Adjacent:** (ones), (ones)

We implemented the noise generation techniques utilizing the open-source toolkits NLTK (Bird et al., 2009) and googletrans to implement the removal of stop words, lemmatization, and backtranslation. For other cases, we implemented noise generation functions from scratch using Python.

Some number of examples of the noisy texts for each noisy type can be found in our GitHub repository mentioned in **section 1**.

We expected that the performance degradation due to the introduced noise for word/grammatical level noises to be less than for character level noises.

7.2 Models

We implemented all models from scratch using pretrained Bert base models from HuggingFace and the PyTorch package. We referred to the official codes of (Karpukhin et al., 2020), (Khattab and Zaharia, 2020), and (Zhuang and Zuccon, 2021), but our codes were mostly written based on the contents written in the original papers. The codes for our models are written in 'model/reranker.py', 'model/dpr.py', and 'model/colbert.py'. We mainly used a desktop that has a GPU with 24 GB of memory. We had one major issue that we couldn't solve which is written in **section 6.7**.

7.3 Result Analysis

The evaluation results are shown in **Table 6**.

Adding word or grammatical noise didn't affect the ranking performance very much. Especially in backtranslation and lemmatize, ranks for most relevant passages seemed to not change or change as little as 1. However, in character level noises such as neighboring character swap or remove space, even with noise in 10% of words made significant performance reduction especially when introduced in the queries. This makes sense considering that in many cases, the WordPiece tokenizer that was used for BERT tokenizes words with character errors into perfectly different sets of tokens. While keywords could appear several times in passages, a keyword being compromised in a short query would be critical to the evaluation performance.

The word/grammatical noise type that affected the results most was the backtranslation for

queries and remove stopwords for passages among the cases shown in Table 6. Overall, the BERT model was robust in word/grammatical errors and we think this is because BERT was pre-trained with much noisy data.

8 Error analysis

For error analysis, we used our baseline models on the reranking task which uses the top 1000 passages based on BM25 for all three models.

8.1 Baseline Error Analysis

We searched for error cases which are when the rank 1 retrieved passage is not a relevant passage based on the Qrels file provided from the official MS Marco website. Within the baseline models, out of a total of 6980 queries, the following number of errors occurred in each model, as well as the number of errors that occurred solely in that model, respectively:

- **Reranker:** 5723, 163
- **DPR:** 5874, 259
- **Colbert:** 5765, 158

4930 queries were commonly error to all three models and for 474 queries, all three models gave a relevant passage as rank 1.

The clearest pattern across all models is that many errors were not really errors at all. There are many cases in which the passage ranked first by the model perfectly suits the query. In some of those cases, the "correct" passage is ranked very low. **Table 7** shows queries that are errors due to the fact that the MS MARCO dataset labels relevant passages as errors. The worst offender might be qid 129641; the query is "define: morbid obesity" and the top model passage is about morbid obesity (though it doesn't quite define it). But the labeled relevant passage is a definition of the word morbidity. This seems like a clear error in the human annotation of the dataset. And as MS MARCO was created by using Bing's passage retrieval system to automatically select on average 10 passages (Nguyen et al., 2016) that may include the answer before having humans annotate these passages, when there's significantly more than 10 valid passages, then many valid passages can fail to be identified as such, leading to examples such as qid 131768 being declared errors when they are not. More examples when the top ranked passages are relevant to the query while they are not included in the qrels file are shown in **Table 7**.

In analyzing the errors, the only categories we could identify are reasonable errors (like the ones discussed above) and unreasonable errors, where the model clearly made a mistake. Examples of unreasonable errors are in **Table 8**. There is not much common thread linking these to each other. Queries like "define scooped out" in example 126821 do stand out as being a bit odd as it's not a phrase that would be commonly searched for. But in our set we weren't able to identify enough errors in similar queries to consider it a pattern. It is worth noting that for qids 1047854, 126821, and 10264, at least one of the models had the correct passage in the top two. This suggests that the errors in these cases are avoidable.

8.2 Approach Error Analysis

In beginning to analyze the errors within our approach, we chose to select the qids of errors that occurred in the models utilizing the noisy query and passage datasets that did not occur in their corresponding baseline model. (i.e. if qid 145 is in error in ColBERT with noisy queries and qid 145 does not show up as an error in the baseline ColBERT model) As such, we are able to isolate the errors that noisy passages and queries produced on the individual models.

When examining **Table 9**, we can see that from the selected queries that all 3 models failed when introduced to a noisy dataset, we can see that the introduced noise seems to distort proper nouns (slackspace vs slack space, Nsetle vs Nestle) or key words (wa vs was) that may have impacted the models ability to retrieve the proper passages.

Similarly, when examining **Table 10** which contains other selected errors, the noise introduced in the queries affected either the proper nouns within the text, or affected interrogative words that are crucial in understanding the question at hand.

One thing to note is that when examining the errors caused by noisy queries and passages, across all three models it was common to see that the correct passage was typically ranked as the 2nd or 3rd most relevant passages, and often times would be within the single digits. It was rare to see egregious errors such that the models ranked the correct passage in the double and triple digits. It is interesting to see that in the presence of noise the models were still able to assign high relevance

to the correct passages, while when the baseline models made errors it was not uncommon to see the correct passage assigned a rank of 700.

In particular, note qid 1098905. In this query, the name of the state Delaware is misspelled, but both ColBERT and DPR still rank the correct passage first. We see this in other examples as well such as qid 1100299, where the name of the drug lotrel is misspelled to ltorel, but two out of three models again have no trouble. It seems that the underlying pretrained BERT model has seen misspellings like delwaare and has learned that delwaare has the same meaning as deleware. **Table 11** shows selected errors caused by the introduction of noise in passages. It was actually somewhat difficult at times to identify specific typos in passages that caused errors in the model. There were very few patterns to be found here; with examples such as qid 266760, having the target passage transformed to be somewhat gibberish clearly gave the model trouble. But even with these examples, two out of three models had no trouble. So it seems likely that because there is some noise in the training data for the underlying BERT models, they can be surprisingly resilient to noise in both query and passage.

9 Contributions of group members

List what each member of the group contributed to this project here. For example:

- Quincy Hughes: wrote the ColBERT part of the proposal/final paper and worked on error analysis.
- Jung Hoon Lim: built and trained models, wrote code for noise generation, acquired evaluation results.
- Dennis Minn: wrote code for the late interaction algorithm in ColBERT and provided organization.
- Christopher Shi: wrote code for noise generation and worked on error analysis.

Everyone participated in weekly meetings and the write-ups for proposal/final paper.

10 Conclusion

It was interesting that we could learn about the problems in information retrieval such as inference time and performance measurements. We could experience some examples of how the encoded vectors calculated using BERT could be utilized.

One of the more surprising difficulties was that transformers take a long time to process data. This factor in combination with the abundance of available text causes very high latency. In many cases, we were required to train or evaluate our models for multiple days or even weeks to receive feedback from our model, especially when we weren't very careful.

Overall, though with lower performance, our models made meaningful outputs, retrieving relevant passages as rank 1 for up to about 18% of queries in the reranking task based on the official dev set. We also have shown that typos of many sorts lead to degradation in performance, but that likely because the corpus that trained the underlying BERT models is huge and contains noisy data itself, these models can be surprisingly resilient. If we could continue working on our project in the future, we would attempt to recreate our original approach of correcting typos.

References

Bagla, K., Kumar, A., Gupta, S., and Gupta, A. (2021). Noisy text data: Achilles' heel of popular transformer based nlp models. *arXiv preprint arXiv:2110.03353*.

Bird, S., Klein, E., and Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Chicco, D. (2021). Siamese neural networks: An overview. *Artificial Neural Networks*, pages 73–94.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., and Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Khattab, O. and Zaharia, M. (2020). Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. (2016). Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Nogueira, R. and Cho, K. (2019). Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, W. X., Dong, D., Wu, H., and Wang, H. (2020). Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2010.08191*.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Robertson, S. and Zaragoza, H. (2009). *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Thakur, N., Reimers, N., Rücklé, A., Srivastava, A., and Gurevych, I. (2021). Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Zhang, Y., Jin, R., and Zhou, Z.-H. (2010). Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1):43–52.

Zhuang, S. and Zuccon, G. (2021). Dealing with typos for bert-based passage retrieval and ranking. *arXiv preprint arXiv:2108.12139*.

qid	query	relevant passage	predicted top ranked passage	ColBERT rank	DPR rank	Reranker Rank
129641	define: morbid obesity	mor ·bid ·i ·ty. n. pl. mor ·bid ·i ·ties. 1. The quality of being morbid; morbidness.	ColBERT: Morbid Obesity is a Serious Health Condition...	732	853	857
131768	definition of a pest	any organism that causes nuisance to man, either economically or medically. Our classification of pests...	Reranker: A definition of pest is: animals causing damage or annoyance to man, his animals, crops or possessions...	113	32	107
1003482	when making a payment what is an aba number	The ACH ABA number, or routing transit number, is a nine-digit number that identifies each bank...	ColBERT: The ABA routing number is a 9-digit identification number assigned to financial institutions by The American Bankers Association (ABA)...	656	246	746
1001926	where are bacteria found?	The flagella of a unique group of bacteria, the spirochaetes, are found between two membranes in the periplasmic space...	DPR: Where are Bacteria Found: Bacteria are found almost everywhere on Earth, including in the seas and lakes, on all continents...	111	288	276
132639	definition of attorney	Definition of LAWYER: A person learned in the law; as an attorney, counsel, or solicitor...	ColBERT: Definition of ATTORNEY. : one who is legally appointed to transact business on another's behalf; especially: lawyer.	86	97	9

Table 7: Baseline Error Analysis: Selected queries that we determined our models to have performed well on despite evaluation metrics saying otherwise. Top ranked passage is the passage that the specified model ranked first. The right three columns denote the rank assigned by each model to the relevant passage. Ellipses indicate truncated passages.

qid	query	relevant passage	predicted top ranked passage	ColBERT rank	DPR rank	Reranker Rank
1047854	what is printing mechanism	Dot matrix printing. Dot matrix printing or impact matrix printing is a...	DPR: Definition of mechanism. 1 1a : a piece of machinery (see machinery 1) ...	2	68	6
126821	define scooped out	scoop something out of something...	ColBERT: Amidst the restaurants and shops crowded along Main Street in Phoenicia, a dessert oasis awaits you inside the Ice Cream Station...	68	8	2
10264	access, how to go to most recent record	In Access desktop databases the GoToRecord macro action has the following arguments...	Reranker: Accessing Your Medical Records. Health Records Online...	1	2	331
134014	definition of discourse analysis pdf	conversational exchanges or written texts. It follows that discourse analysis is also...	Reranker: Definition of Discourse. Discourse is any written or spoken communication...	222	124	26
1091807	actress who played rosario	Prime-time television has also been known to feature Latina actresses as maids. Some examples have been Rosario on Will & Grace played by Shelley Morrison...	ColBERT: Kapamilya Actress Arci MuÃ±oz revealed that she and boyfriend Badi del Rosario are no longer together...	21	124	12

Table 8: Baseline Error Analysis: Selected queries that one or models failed to identify the relevant passage for. Top ranked passage is the passage that the specified model erroneously ranked first. The right three columns denote the rank assigned by each model to the relevant passage. Ellipses indicate truncated passages.

qid	query and noise type	relevant passage	top ranked passage	ColBERT rank	DPR rank	Reranker Rank
921173	what was the length of the titanic (Passages Lemmatize 100%)	The length of the Titanic wa 882 foot and 9 inch (that's about 269 meter). The ship wa 92 foot and 6 inch (28 meter) wide and 175 foot (53 meter) high.	DPR: How many pound or ton did the titanic weigh. 1 882 foot 9 inch - the length of the Titanic (269.1 metres)...	9	4	3
796812	what is slack space (Passages Remove 10% Spaces)	slackspace is the unusedspace between theend...	ColBERT: Before going to explain slack space, one should know what-blocks (on Linux) and clusters mean (on Windows). Blocks are specific sized containers used by file system to store data.	40	35	13
1053611	what is in a chunky bar (Passages Neighbor Char Swap 10%)	Nestle Chunky. Nsetle Chunky si a candy bar known for its trapezoidal shape...	Reranker: Chunky Single Candy Bars, 24 Count Box (Pack of 2) is a candy bar that is a big chunk of chocolate with peanuts and raisins. Each bar can be brkoen inot 4 parts. Product details	2	3	2

Table 9: Approach Error Analysis: Selected queries that all three models failed to identify the relevant passage for. Top ranked passage is the passage that the specified model erroneously ranked first. The right three columns denote the rank assigned by each model to the relevant passage. Ellipses indicate truncated passages.

qid	query and noise type	relevant passage	top ranked passage	ColBERT rank	DPR rank	Reranker Rank
1097069	avergae gas costs in kentucky (Queries Neighbor Char Swap One)	The average cost of gas for that typical customer is $61.70. The projected...	ColBERT: Wholesale costs this year are, on average, only 2 percent higher than a year ago. Four of Kentucky's...	2	2	1
1098905	how is delwaare sales tax calculated (Queries Neighbor Char Swap One)	How 2018 Sales taxes are calculated in Wilmington. The Wilmington, Delaware...	Reranker: Figure 1 shows how the sales tax is calculated if a retailer sells five books costing $20 each and the tax rate is 8 percent...	1	1	83
1094962	how htick is the concrete in a garage floor? (Queries Neighbor Char Swap One)	The standard concrete garage floor is 15 cm (6 inches) thick and, with use of proper control joints, should be thick enough to withstand normal use.	ColBERT: Concrete truck can't access driveway, pour options. My back yard is generally...	15	3	1
171776	does strepcause fever blisters (Queries Remove One Space)	It's hard for me to comment on what all is going on. depends on how she was tested for strep and also...	Reranker: old sores (aka fever blisters) are small red blisters that usually appear on the lips and outer edges of the mouth. They often...	2	1	4
1100299	generic ltorel side effects (Queries Neighbor Char Swap One)	Lotrel (amlodipine/benazepril) IS a drug prescribed for the treatment of high blood pressure...	DPR: The common side effects of Generic for Aggrenox* are headache, nausea, vomiting, diarrhea, stomach pain, heartburn, and drowsiness...	1	31	1

Table 10: Approach Error Analysis: Selected noisy queries that one or more models failed to identify the relevant passage for. Top ranked passage is the passage that the specified model erroneously ranked first. The right three columns denote the rank assigned by each model to the relevant passage. Ellipses indicate truncated passages.

qid	query and noise type	relevant passage	top ranked passage	ColBERT rank	DPR rank	Reranker Rank
266760	how long should i boil corn on the cob? (Passage Word Order Swap)	In pot, large a enough to hold the corn, fill it with water to cover the corn (the corn should float). the a medium heat...	ColBERT: Steps to Boil Corn on the Cob: 1 1) Fill a large pot about half full of water and bring to a boil. 2 Husk...	18	1	1
869035	what kind of leg pain comes from sciatica (Passage Word Order Swap Adjacent)	in is a shooting pain that begins Sciatica the lower back, radiates into the buttock...	ColBERT: Sciatica is leg pain back by a pinched nerve in the lower caused. Although the pangs begin in nerve roots located...	91	1	1
422268	is rayon same as cotton? (Passage Character Swap)	exsits and si an alternate of . No, cotton adn rayon are entirely different. Cotton si a material that comes from the cotton plant...	DPR: What is rayon? Unlike cotton thread which is made from a natural source and unlike polyester which is made from man-made polymers, rayon (commonly...	1	20	1
541948	watery blurry eyes (Passage Remove 50% Stopwords)	Other Eye Conditions: For example, glaucoma, cataracts, or macular degeneration. Dry Eyes: Blurry vision symptom this syndrome...	Reranker: Watery eyes is most common sign a blocked tear duct. Tears spill over onto your cheek. They may thicker...	1	1	69

Table 11: Approach Error Analysis: Selected noisy passages that one or models failed to identify the relevant passage for. Top ranked passage is the passage that the specified model erroneously ranked first. The right three columns denote the rank assigned by each model to the relevant passage. Ellipses indicate truncated passages.

qid	query	relevant passage	top ranked passage
2235	Bethel University was founded in what year	Bethel University is a private institution that was founded in 1842. It has a total undergraduate enrollment of 4,792, its setting is rural, and the campus size is 100 acres. It utilizes a program-based academic calendar.	Bethel University is a private institution that was founded in 1871. It has a total undergraduate enrollment of 3,051, its setting is suburban, and the campus size is 289 acres.
130932	definition for naive	1. naive-marked by or showing unaffected simplicity and lack of guile or worldly experience; a teenager's naive ignorance of life; the naive assumption that things can only get better; this naive simple creature with wide friendly eyes so eager to believe appearances.	Definition of naive for English Language Learners. : having or showing a lack of experience or knowledge : innocent or simple.
132639	definition of attorney	Definition of LAWYER: A person learned in the law; as an attorney, counsel, or solicitor. Any person who, for fee or reward, prosecutes or defends causes in courts of record or The Law Dictionary Featuring Black's Law Dictionary Free Online Legal Dictionary 2nd Ed.	Definition of ATTORNEY. : one who is legally appointed to transact business on another's behalf; especially: lawyer.
1001926	where are bacteria found?	The flagella of a unique group of bacteria, the spirochaetes, are found between two membranes in the periplasmic space. They have a distinctive helical body that twists about as it moves.	Where are Bacteria Found: Bacteria are found almost everywhere on Earth, including in the seas and lakes, on all continents (including Antarctica), in the soil, and in tissues of plants and animals.nlike animals and plants, bacteria have pili, flagella, and most have a cell capsule. Bacterial cells include the following: Diet: Bacteria have a wide range of diets. Some are heterotrophs (they eat other organisms) and others are autotrophs (they make their own food).
36965	average heavy equipment mechanic salary	Heavy Equipment Mechanic average salary is $34,143, median salary is $31,512 with a salary range from $22,880 to $46,488.	Heavy Equipment Mechanic Salary. Heavy Equipment Mechanic average salary is $34,143, median salary is $31,512 with a salary range from $22,880 to $46,488.

Table 12: Examples of the predicted result where the model failed to predict the relevant passage as rank 1, but the rank 1 passage seem relevant to the query despite it is not shown in the qrels file. In the case of qid with 130932, the relevant passage from the qrels file doesn't actually seem relevant.