Contrastive Representation Learning for Exemplar-Guided Paraphrase Generation

Advanced Natural Language Processing (Monsoon 2024)

Final Submission

Rage against the Machine Learning (Team 33)
Bassam Adnan (2023121003)
Samyak Mishra (2022101121)
Varun Edachali (2022101029)

November 20, 2024

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY

HYDERABAD

1 Problem Statement

In this project, we explore a new method to generate paraphrases of source sentences, using other sentences, called exemplars, as style guides (Exemplar-Guided Paraphrase Generation), proposed by Yang et al.[1] propose a Contrastive Representation Learning approach for learning content and style information and using them to generate paraphrases. The authors introduce two contrastive learning objectives to the model via two contrastive losses, one for content information and the other for style information, and by minimising these in addition to the usual generation loss, the model learns to paraphrase sentences in the exemplars' style.

In this project, we implement the author's approach on our own, using their implementation as a baseline, and evaluate our results on the Quora Question-Pairs dataset, `QQP-Pos`. In addition, we perform ablation studies by replacing the main learning components of the encoders and decoders in the model with suitable alternatives, and perform hyperparameter testing by varying the contribution of the content and style contrastive losses. We report our results and present some analyses based on them.

2 Selecting Baselines from the Literature Study

Since our project is entirely focused on the CRL-EGPG paper, involving a from-scratch implementation and an ablation study, we used their reported scores as the baseline for our project. We compare the results of evaluating our implementation against theirs, and achieved almost as good results.

3 Dataset

The dataset used extensively throughout our testing is `QQP-Pos`. The original Quora Question Pairs (`QQP`) dataset contains about $400k$ sentence pairs labeled positive if they are duplicates of each other and negative otherwise. If we select those positive pairs that contain both sentences with a maximum token length of 30, we are left with the `QQP-Pos` dataset (source).

Thus, this dataset can be readily used for paraphrase generation, as it contains questions that have been identified as duplicates of each other.

The paper also mentions the `ParaNMT` dataset, which contains explicit sentence-paraphrase pairs. However, this dataset is quite vast, and we did not have the compute to run even a single train loop on it. Considering our system of choice, `Kaggle` has a 12-hour runtime limit and `QQP-Pos` took about 7 hours to run the baseline parameters while being less than a third of the former's size - we limited our analysis of this dataset to processing and running a few epochs for validation of our model.

4 Approach

In the paper, the authors propose a new method to learn better representations of style and content information from the exemplar and source sentences respectively. They suggest using two contrastive losses, one each for the content and style information, and weighing in their contribution in addition to the regular generation loss.

In their proposed model, they use an encoder each to extract the content and style information.

$$c_{X_i} = E_c(X_i)$$

$$s_{X_i} = E_s(X_i)$$

The model then concatenates these two to use as the initial hidden state $h_0 = [c_{X_i}, s_{X_i}]$ of a GRU block in the decoder module, which they use to iteratively generate the output sentence.

To compute the content contrastive loss (CCL), they use the target sentence Y_i corresponding to the source X_i, i.e. they contrast c_{X_i} with c_{Y_i}, used to minimize the distance between positive pairs and maximise that between negative pairs. Similarly for the style contrastive loss (SCL), the exemplar Z_i's style s_{Z_i} is contrasted with that of the source, s_{X_i}.

The base optimisation objective is taken to be the negative log-likelihood (NLL) loss of the model output against the label target sentence Y_i, and add to it the contrastive losses weighed by parameters λ_1 and λ_2 respectively.

The combined two-encoder, one-decoder model is then trained on the dataset for 45 epochs, using the Adam optimisation algorithm. Notably, the contrastive losses are only used against the train set, and the validation set is used only to calculate the NLL loss and perplexity.

5 Our Implementation

We used the authors' implementation as a guide for the project, and wrote up the code from scratch, in PyTorch.

5.1 Overall Architecture

The model takes in a source sentence and a style sentence, and predicts a paraphrase of the source sentence in the style of the exemplar. It consists of two encoders: one for content (of the source sentence) extraction and one for style extraction (of the exemplar sentence), and a decoder to generate the paraphrases. The content encoder and decoder are packaged together in one model, called `Seq2Seq`, and the style encoder, in a model called `StyleExtractor`. We train both the `Seq2Seq` and `StyleExtractpr` models on the dataset, with the hyperparameters and ablation parameters read from a config file, and save the trained models. We then process the models on the test set, to prepare text files for calculating scores. We compare the model's generated output to the target sentences and calculate ROUGE, BLEU and **OTHER** scores.

5.2 Preparing the Dataset

The processing script can be used to extract relevant information from the `QQP-Pos` and `ParaNMT` datasets. We store the following information:

- *word-to-index* and *index-to-word* mappings, essentially building the vocabulary. Note that the PoS tags are also included in this vocabulary.
- For each set (train, test and validation, both source and target) store the tokens and corresponding parts-of-speech tags corresponding to each sentence.
- Store the bert-ids corresponding to each token for each sentence in the train, test and validation sets - corresponding to a pre-trained BERT model.
- Store similar sentences for each sentence in the target sets by style. More specifically, store upto 5 sentences that have a minimum edit distance from our current sentence in terms of their parts-of-speech tags, while ensuring they maintain some arbitrary bounds regarding their lengths and the number of common words they share.

During training, we take the source sentence to paraphrase (the model should learn the semantics / content from here) and a sentence from the similarity list of the target sentence (the model should learn the syntax / style from here) in order to generate the target sentence. An example of this inference from the quora dataset is as below:

Source Sentence: can you suggest a best budget phone below 15k ?

Similar Sentences (to Target):

- which mobile is better under 15k ?
  tags: ['WDT', 'NN', 'VBZ', 'RBR', 'IN', 'CD', '.']
- which bicycle should i buy under 10k ?
  tags: ['WDT', 'NN', 'MD', 'VB', 'VB', 'IN', 'CD', '.']

Target Sentence:
which phone is best to buy under 15k ?
tags: ['WDT', 'NN', 'VBZ', 'RBS', 'TO', 'VB', 'IN', 'CD', '.']

Note that the "tags" are the index_to_word mappings of the PoS tags of the respective sentences. For example, *NN* denotes a singular noun and *VBZ* denotes a verb.

Notice the similarity in content to the source, and the similarity in style (quantified by edit distance in parts-of-speech tags) to the similar sentences, shown in the target. Thus, the similarity list is integral to the training loop.

NOTE: the number of sentence-paraphrase pairs in the datasets are as below:

	QQP-Pos	ParaNMT
train	137185	493081
test	3000	800
valid	3000	500

The training was done on the `QQP-Pos` dataset due to computational constraints (we used `Kaggle` to train our models, which has a 12-hour cap on runtime). Qualitatively, the data in `QQP-Pos` seemed better and, in the words of the paper, 'more formal' than `ParaNMT` anyway.

5.3 Style Extractor

The paper proposes a `BERT` based style encoder to extract style features from the exemplar. Our implementation supports easy ablation between `BERT`, `ALBERTA` and `RoBERTa` for extensibility and experimentation.

5.4 Sequence-to-Sequence Model

5.4.1 Encoder

The encoder model is any sequential model whose task is to encode the content / semantics of the source sentence. Our implementation allows for simple swapping of the encoder type among `RNN`, `GRU` and `LSTM` types.

5.4.2 Decoder

The decoder model is essentially a wrapper for a sequential model that generates a target sentence using the output of the content encoder (semantics) and of the style encoder (syntax) as its initial hidden state. As with the `seq2seq` encoder, we have allowed to choose an `RNN`, `GRU` or `LSTM` for the decoder.

5.5 Training

We train the model (and its ablated versions) on the `QQP-Pos` dataset over 45 epochs using the Adam optimiser, using the contrastive losses with the train set (in addition to the base NLL loss), and only calculating NLL loss (and perplexity) to evaluate it on the validation set. For each epoch, we store the averages of these losses, in separate files for the training and validation losses.

5.6 Evaluating

5.6.1 Preparing Model Output

Essentially, we run inference on the test set. Here, however, instead of selecting a random similar sentence of the target sample as our exemplar, we generate a paraphrase for the source using each of the similar sentences. We then select the exemplar whose paraphrase provides the best coverage with the target sentence.

The generated paraphrase as well as the exemplar corresponding to each sentence in the test set are stored in separate files to be fed further down the evaluation pipeline.

5.6.2 Evaluating the Output

The paper itself uses SGCP[2]'s code to evaluate the output. We have used a modified version of the same, that takes in the outputs of the previous step (the generated paraphrases and exemplars) and uses it to calculate a number of metrics such as `BLEU` score, `S-TED` and so on.

6 Results

Table 1: Baseline (Authors' implementation of CRL-EGPG)

BLEU	R-1	R-2	R-L	METEOR	S-TED	T-TED
45.400	70.518	52.063	72.734	45.095	6.040	7.656

Table 2: Our implementation of CRL-EGPG

BLEU	R-1	R-2	R-L	METEOR	S-TED	T-TED
44.730	69.991	51.448	72.474	44.283	5.998	7.834

By default, we used $\lambda_1 = \lambda_2 = 0.1$ as the weights for our content and style losses, `GRU` as the sequential model in our content encoder, and decoder. For the style-extractor, we used a `BERT` model and extracted the output from the last layer. This is to align with the parameters chosen by the original authors.

6.1 Seq2Seq Variation

Table 3: Seq2Seq Variation Comparison

Encoder	Decoder	BLEU	R-1	R-2	R-L	METEOR	S-TED	T-TED
GRU	LSTM	44.880	72.842	53.085	74.455	45.787	6.096	8.007
GRU	RNN	44.090	71.422	51.969	73.362	45.085	6.570	8.515
LSTM	GRU	45.230	70.440	51.961	72.709	45.383	6.092	7.838
RNN	GRU	42.590	68.284	49.758	70.890	42.459	6.086	7.502
RNN	RNN	42.040	70.180	50.339	72.208	43.356	6.612	8.272

We varied different Encoder and Decoder models for our Content model. Originally a pair of GRU's were used, we were interested to see how changing of these components affected the scores. They show an interesting variation. LSTM-GRU even come close to our original model performance.

Note: we could not run the model with an `LSTM` encoder and decoder due to the entire 12-hour timeout running out during training.

6.2 Style-Extractor Model Variations

6.2.1 Whole Model Ablation

Table 4: BERT Model Comparison

Model	BLEU	R-1	R-2	R-L	METEOR	S-TED	T-TED
ALBERT	46.530	71.486	53.334	73.533	46.547	6.050	8.505
RoBERTa	46.680	71.650	53.181	73.721	46.579	5.916	8.361

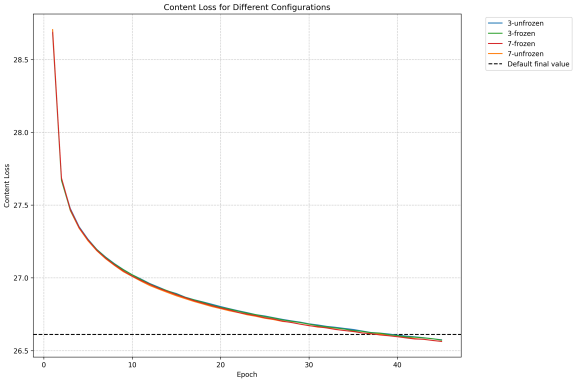
Using the last layer of ALBERT and RoBERTa for the Style embeddings improved the performance across most metrics compared to the default method ($\lambda = 0.1$).

6.2.2 BERT Layer Configuration

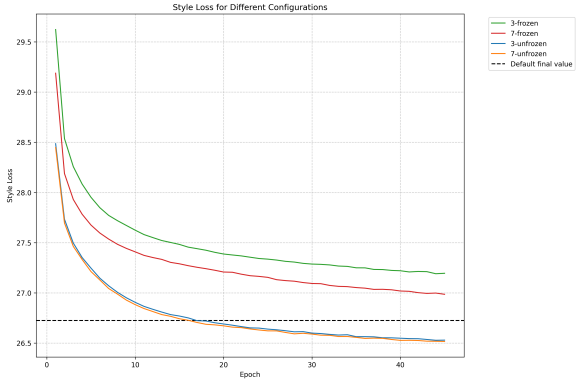
Table 5: BERT Layer Configuration Comparison

Layer	Training	BLEU	R-1	R-2	R-L	METEOR	S-TED	T-TED
3	Freeze	43.200	70.390	50.631	72.414	44.602	6.379	7.789
3	Unfreeze	42.870	70.180	50.478	72.226	44.370	6.292	7.709
7	Freeze	43.070	69.914	50.129	71.993	44.031	6.318	7.683
7	Unfreeze	42.830	70.135	50.295	72.247	43.902	6.359	7.668

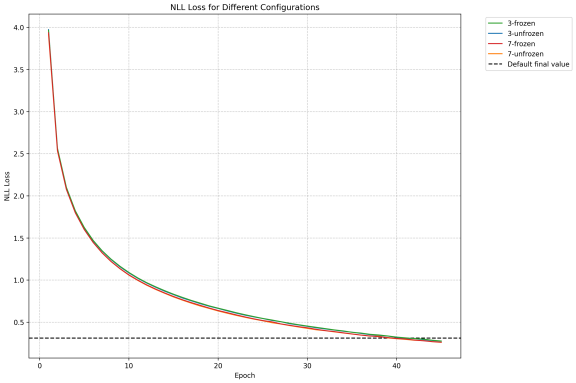
Variation of the metrics when using different layers of BERT $(3, 7)$ with different modes of training. In Freeze training, we freeze every proceeding layer. We train the entire network in Unfreeze. We were motivated to try this since the earlier layers of BERT would have already encoded rich linguistic syntactic features.



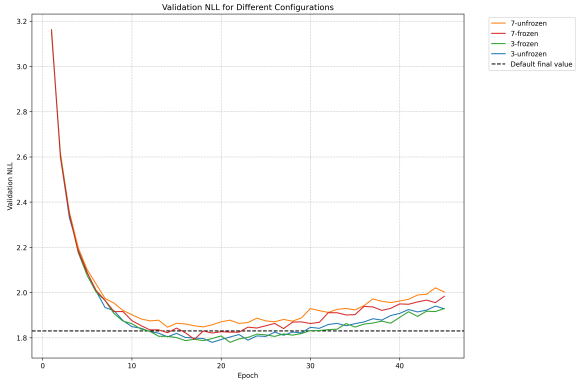
(a) Content Loss



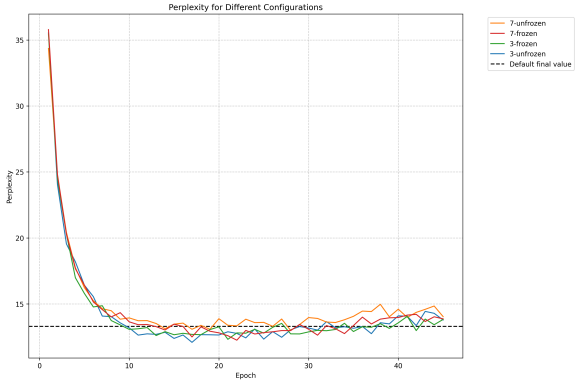
(b) Style Loss



(c) NLL Loss



(d) Validation NLL



(e) Perplexity

Figure 1: Analysis of Different BERT Layer Configurations

6.3 Further Experiments

6.3.1 λ Variation

Table 6: λ Comparison ($\lambda_1 = \lambda_2$)

λ	BLEU	R-1	R-2	R-L	METEOR	S-TED	T-TED
0.001	40.820	68.683	48.315	70.940	42.030	6.440	7.480
0.01	42.060	69.303	49.377	71.526	42.688	6.218	7.395
0.1	45.400	70.518	52.063	72.734	45.095	6.040	7.656
0.3	47.340	71.657	53.786	73.848	47.036	5.904	7.863
0.5	46.940	71.544	53.612	73.671	46.819	6.041	7.961
0.7	47.040	71.452	53.571	73.641	46.794	6.018	7.962
1.0	46.640	71.187	53.314	73.343	47.168	6.075	8.114
1.3	46.640	71.272	52.986	73.315	47.131	6.198	8.172
1.5	45.850	70.912	52.624	73.001	46.442	6.293	8.189
1.7	46.250	70.711	52.794	72.821	46.878	6.046	8.008
2.5	45.370	70.288	52.039	72.426	45.955	6.098	7.977

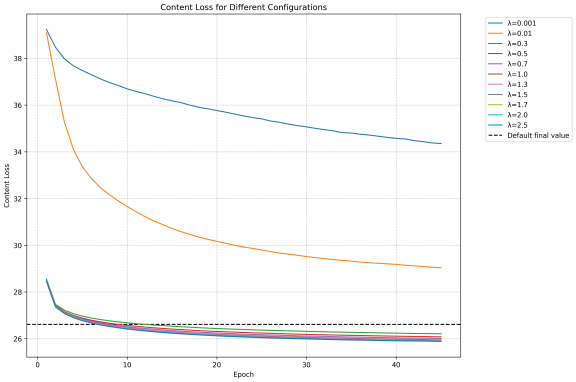
We varied λ values and observed satisfactory results for the range $0.3-0.7$. The impact of λ starts to diminish as its increased, although its significance is not trivial since without the content/style loss the scores are much less.

6.3.2 Varying λ_1 and λ_2

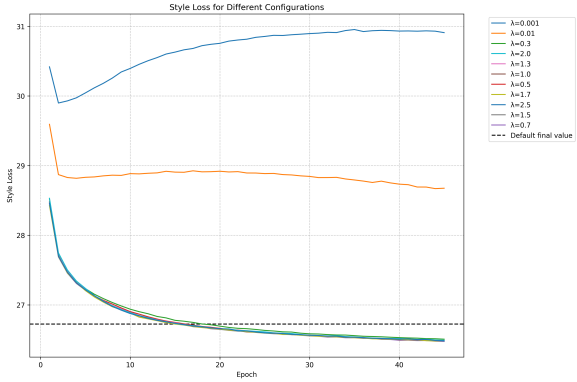
Table 7: Different λ_1 and λ_2 Comparison

λ_1	λ_2	BLEU	R-1	R-2	R-L	METEOR	S-TED	T-TED
0.01	0.1	38.880	69.328	49.715	71.569	35.746	12.759	11.797
0.1	0.01	38.220	68.774	49.293	71.118	35.347	12.794	11.757
0.1	0.1	44.730	69.991	51.448	72.474	44.283	5.998	7.834
0.1	0.2	45.420	70.273	52.198	72.573	45.280	6.024	7.914
0.1	0.3	46.270	70.814	52.680	73.056	46.220	6.006	8.037
0.1	0.4	45.390	70.227	52.135	72.524	45.300	6.088	8.055
0.2	0.1	44.920	70.152	51.687	72.526	44.929	6.017	7.690
0.3	0.1	45.130	70.322	51.856	72.625	45.201	5.941	7.762
0.4	0.1	44.640	69.406	51.266	71.765	44.592	6.180	7.895

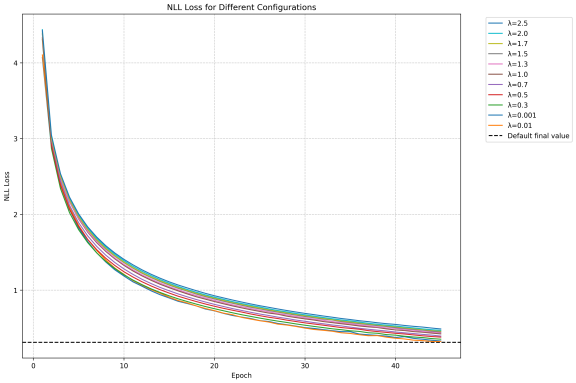
The variation of λ_1 and λ_2 does not seem to vary the metrics by a great deal. However reducing the impact of any one of the λ to 0.01 does affect the BLEU metric significantly.



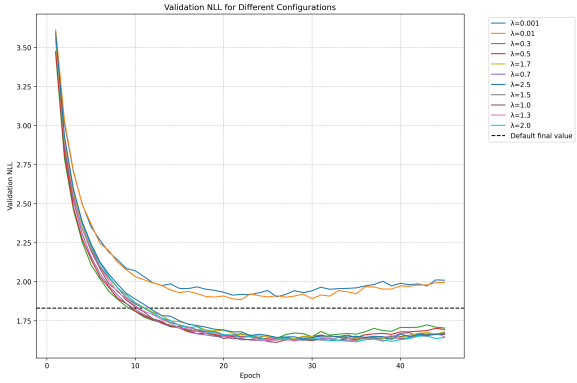
(a) Content Loss



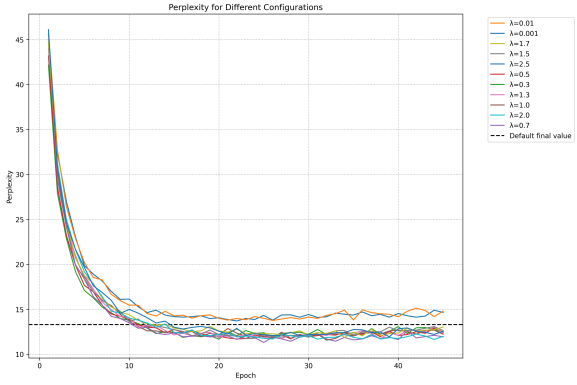
(b) Style Loss



(c) NLL Loss

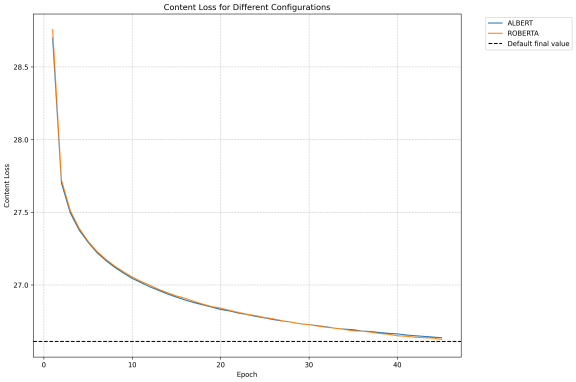


(d) Validation NLL

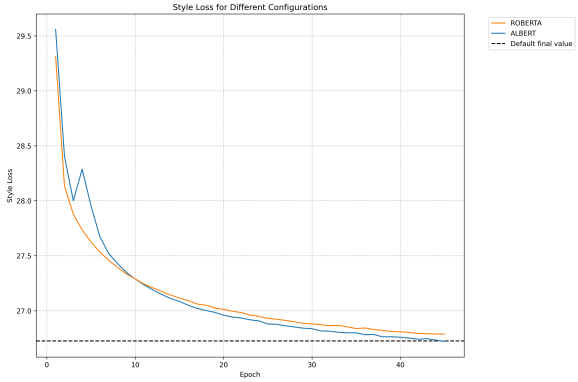


(e) Perplexity

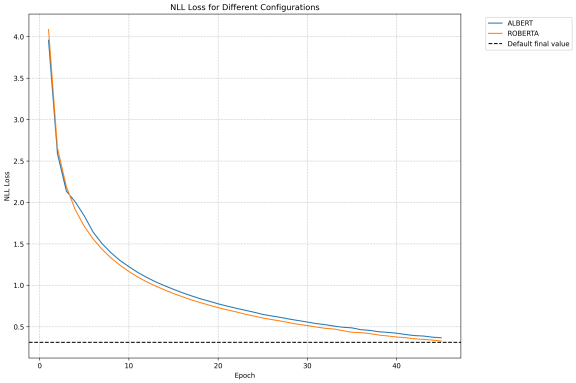
Figure 2: Analysis of Different λ Values



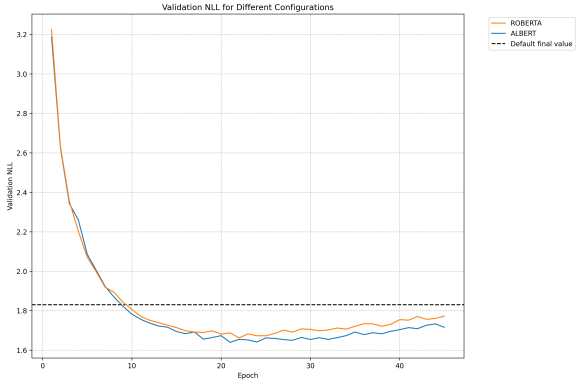
(a) Content Loss



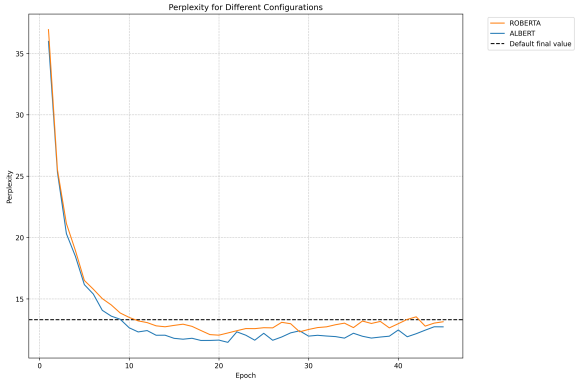
(b) Style Loss



(c) NLL Loss



(d) Validation NLL



(e) Perplexity

Figure 3: Analysis of Different BERT Models

6.3.3 Quantisation

Table 8: Quantization Comparison

Model	BLEU	R-1	R-2	R-L	METEOR	S-TED	T-TED
Full Quantization	44.660	70.333	51.568	72.631	44.587	6.099	7.878
Selective Quantization	45.610	70.757	52.372	72.970	45.343	6.019	7.717

Quantization did not seem to affect the results. In selective quantization, we kept the original projection layer in our Content Encoder/Decoder unchanged. The entire model quantization (to "int8" datatype) resulted size reduction by 54.53%.

Input	Exemplar	Output	Description
how do i develop good project management skills?	what helps you pass a meth test?	what helps you build up management skills?	Preserves query intent while adopting exemplar's structure (Our model output on $\lambda_1 = \lambda_2 = 0.1$)
which is the best anime to watch?	what helps you pass a meth test?	what is best as you recommend watch?	Preserves query intent while adopting exemplar's structure although it tries to fit in words (What/Why) probably to minimize the loss during training (Our model output on $\lambda_1 = \lambda_2 = 2.0$)
	why do some people like cats more than dogs?	why are some good anime for watching ?	
how do i develop good project management skills?	what is the easiest way to make a little money online?	development skills?	7'th BERT Layer output used for the style encoding is unable to encode the structure of exemplar.
how do i develop good project management skills?	why do some people like cats more than dogs?	how do i have improved project management for beginners?	Results for $\lambda_1 = \lambda_2 = 0.001$, this is the output when the application of contrastive loss is minimized.
	what is the easiest way to make a little money online?	what are some good way to develop a public project management skills?	

Table 9: Qualitative analysis of style transfer examples (Part 1): Base model and hyperparameter variations

7 Analysis

7.1 Qualitative Analysis

We test the model with random sentences (user input) against some exemplars to see what the model is generating. We display the examples in two tables: one using the base model and hyperparameter variations, and the other using further hyperparameter variations and ablations.

8 Conclusion

Since our mid submission, we have implemented the paper from scratch, using the autohors' implementation as a guide. We have trained the model on the Quora dataset with the default configurations as a baseline, and our results match the authors' reported scores. As ablation studies, we have swapped out the BERT encoder block in the style extractor with RoBERTa and ALBERT, and the GRU component of the content encoder and decoder with an LSTM, and have evaluated the subsequently trained models. We have also performed hyperparameter testing on the contrastive loss weights, λ_1 and λ_2 to assess the impact of one loss over the other. We attempted to replace the GRU components of the content encoder and the decoder with respective encoder and decoder blocks, but decided not to continue with it, as the architecture would then become too different to be considered an ablation.

We have presented our results in this report, and have linked our repositories and notebooks used for the project.

8.1 Links

- GitHub Organisation - Link
- QQP Training Notebook - Link
- ParaNMT Training Notebook- Link

Input	Exemplar	Output	Description
how do i develop good project management skills?	why do some people like cats more than dogs?	how do some students create good ideas skills ?	Results for $\lambda_1 = 0.1, \lambda_2 = 0.2$, Giving higher λ value to style loss did not help encoding the meaning of the sentence and gave more focus to the style.
	what helps you pass a meth test?	what skills to create an engine marketing skills ?	
how do i develop good project management skills?	why do some people like cats more than dogs?	how do a business projects help me good job jobs ?	Results for $\lambda_1 = 0.2, \lambda_2 = 0.1$, Giving higher λ value to content loss did not help in preserving the style of the exemplar although still encodes meaning to an extent and gave more focus to the content.
	what helps you pass a meth test?	what skills should a good project manager?	
how do i develop good project management skills?	what are some safe and legal ways to view a private facebook profile?	what are some good and tricks learning resources to improve my project management?	LSTM-GRU output.
	what helps you pass a meth test?	what are the best ways to develop good project management skills ?	GRU-RNN output

Table 10: Qualitative analysis of style transfer examples (Part 2): Further hyperparameter tuning and architectural variations

- Dataset - Link
- **Style Encoder Ablations:**
  1. RoBERTa: Link
  2. ALBERT: Link
- **Content Encoder + Decoder Ablations:**
  1. GRU Encoder + LSTM Decoder: Link
  2. LSTM Encoder + GRU Decoder: Link
  3. RNN Encoder + GRU Decoder: Link
  4. GRU Encoder + RNN Decoder: Link
  5. RNN Encoder + RNN Decoder: Link
- **Content Encoder + Decoder Ablations:**
  1. BERT Layer 3 (Frozen): Link
  2. BERT Layer 3 (Unfrozen): Link
  3. BERT Layer 7 (Frozen): Link
  4. BERT Layer 7 (Unfrozen): Link
- **Contrastive Loss Weights Hyperparameter Testing:**
  - Both weights identical:
    1. $\lambda = 0.001$: Link
    2. $\lambda = 0.01$: Link
    3. $\lambda = 0.3$: Link

4. $\lambda = 0.5$: Link
5. $\lambda = 0.7$: Link
6. $\lambda = 1.0$: Link
7. $\lambda = 1.3$: Link
8. $\lambda = 1.5$: Link
9. $\lambda = 1.7$: Link
10. $\lambda = 2.0$: Link
11. $\lambda = 2.5$: Link

– Varying style loss weight while keeping content loss weight constant

1. $\lambda_1 = 0.1$, $\lambda_2 = 0.2$: Link
2. $\lambda_1 = 0.1$, $\lambda_2 = 0.3$: Link
3. $\lambda_1 = 0.1$, $\lambda_2 = 0.4$: Link

– Varying content loss weight while keeping style loss weight constant

1. $\lambda_1 = 0.2$, $\lambda_2 = 0.1$: Link
2. $\lambda_1 = 0.3$, $\lambda_2 = 0.1$: Link
3. $\lambda_1 = 0.4$, $\lambda_2 = 0.1$: Link

8.2 Submission

Our zip file contains the following:

- `CRL-EGPG-From-Scratch`: The codebase where we implement the paper on our own, containing a clean implementation of the complete model
- `NLP-EVAL`: Standard code to evaluate the model.
- `REPORT.pdf`: Well, our report (this document).

References

[1] H. Yang, W. Lam, and P. Li, "Contrastive representation learning for exemplar-guided paraphrase generation," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 4754–4761. [Online]. Available: https://aclanthology.org/2021.findings-emnlp.409

[2] A. Kumar, K. Ahuja, R. Vadapalli, and P. Talukdar, "Syntax-guided controlled generation of paraphrases," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 329–345, 2020. [Online]. Available: https://aclanthology.org/2020.tacl-1.22