Model Explorations in Abstractive Summarisation

Youning Xia 19054254
Yan Song 9898418
Shuyi Han 19060915
Wan Jing Song 17130843

Abstract

This paper sets out to assess the performance of Deep Reinforcement Learning (DRL) based abstractive summarization models. 4 different model variants are applied on 3 datasets and evaluate on ROUGE and BERTScores. Working on the novel WikiHow dataset (Koupaee and Wang, 2018) which is slightly more complex to train on has magnified the characteristics of the models. It exposes the instability of training on ROUGE-L scores and suggest BERTScore as an alternative.

1 Introduction

Text summarization is often viewed as a fundamental skill in language processing. Especially with the increasing speed of information generated in current times; the ability to summarize is all the more useful. There are two main summarization styles—Extractive Summarization and Abstractive Summarization (AS).

In extractive summarization, the model evaluates chunnks within the corpus; concatenating them as a summary. The challenge lies in selecting points. Extracted summaries risk being biased if comprised of similar strongly worded points (Barzilay et al., 1999). AS goes further by *paraphrasing* points more concisely. This requires not only understanding of the input but also preservation of ideas in the output. Accordingly, abstractive summaries outperform extractive ones where the corpus contains diverse and controversial points (Cheung, 2008).

Text transformation involve natural language generation (NLG). seq2seq problems saw breakthroughs with Long Short-term Memory (LSTM) models (Graves, 2013). Further developments occurred in the field of machine translation by combining LSTM with encoder-decoder (Sutskever et al., 2014). A second crucial progress came from Bahdanau et al. (2014)'s Attention model that could direct focus on input sequences via context vectors. This builds context more efficiently and comes handy for summarization tasks in highlighting keywords within input text.

While summarization and translation are closely related in the need to preserve meaning, they have differing expectations (Nallapati et al., 2016). Translated text are expected to preserve exact meaning without restriction on output length. Whereas summaries are bounded by length with some leeway in interpretation. The compression process divides the two tasks.

Early extractive approaches were focused on manually feature engineering extractive techniques. Riding the trend towards neural network engineering, models seen in Cheng and Lapata (2016), Chopra et al. (2016) were designed for models to learn features themselves.

The following year, Paulus et al. (2017) fit the AS task in a Reinforcement Learning (RL) context. Since then, there has been greater interest in using RL for AS. Main exploration has been in tuning reward settings like in Li et al. (2019).

Apart from improvements in model architectures, the field has also inspected the nature and quality of datasets. For a long time, popular datasets used in training and evaluation circled around news-based datasets like the NYT and CNN/DailyMail datasets. However, these tend to follow a certain style (e.g. Inverted Pyramid style (Koupaee and Wang, 2018)) which if picked up by models may not perform well on general passages. Accordingly, challenging and high quality datasets were developed to augment variation.

2 Related Works

Initial headway into AS emerge from RNN models that were popular with seq2seq problems.

As NLP techniques expanded, architectural features like encoder-decoder (Sutskever et al., 2014) and attention (Bahdanau et al., 2014) frameworks proved useful in boosting AS performance (Nallapati et al., 2016), (Chopra et al., 2016).

However, it was observed that during the decoding process, error could accumulate and have models prone to exposure bias (Li et al., 2019). One way to manage this bias is RL where corrective feedback (against bias) is provided to the models.

The formulation of different metrics like BLEU and ROUGE to evaluate sequence output made it possible to frame text generation models in RL settings. Ranzato et al. (2015) adapted the REINFORCE algorithm and cross entropy as the MIXER algorithm and applied it on some text generation tasks including summarization. Bahdanau et al. (2017) modified the temporal difference method using ground-truths to train a *critic* RNN and the main prediction *actor* network.

Paulus et al. (2017) included elements of both conventional NLP models and RL. Intra-temporal attention was applied on the LSTM encoder combined with a new intra-decoder attention mechanism. Two RL strategies were applied to train the encoder-decoder network. A mixed training objective function, consisting teacher forcing and policy learning objectives, was used to train the model. In isolation, supervised learning with teacher forcing though able to generate more fluently, can have exposure bias since it has access to ground-truths. The self-critical policy gradient training algorithm mitigates this.

Chen and Bansal (2018) used RL in a slightly different manner. The paper's focus was on exploiting the advantages of both extractive and abstractive summarization tasks; and RL to link the two. The extractive network identifies key sentences, passes these output sentences into the abstractive network to be further summarized. This notion is similar to early extract-then-compress methods (Jing and McKeown, 2000). Advantage Actor-Critic algorithm was implemented with the extractor network as the agent. Because RL is done at the sentence-level without distubring abstraction, fluency is preserved.

The BERTScore by Zhang et al. (2019), unlike BLEU and ROUGE, captures semantic relationships between output sentences. By comparing contextual embeddings rather than exact matching, BERTScore detects better for meaning at the same time allowing greater output diversity. Li et al. (2019) uses BERTScore as a Distributional Semantic Reward (DSR) quantifier in the reward objective. Resulting models generated passages that were more concise and fluent compared to those under BLEU and ROUGE.

In this paper, we explore 4 model variants on 3 different datasets to investigate the performance of RL models against models with different training objectives. In addition, testing the performance of reward functions ROUGE-L against DSR-type BERTScore.

3 Models

The model replicated in this paper is largely modelled on Paulus et al. (2017) with complementary features from See et al. (2017). It is an end-to-end model with intra-attention applied on both encoder and decoder, along with a pointer mechanism to generate the output, illustrated in Figure 1. In the following equation, we denote the input sequence of tokens as $\{x_1, x_2, ..., x_m\}$ and output summarisation sequence as $\{y_1, ..., y_n\}$.

3.1 Encoder-decoder Network

The encoder consists of a bi-directional LSTM which reads the embedding vector of input x_i. Forward and backward hidden states $[\boldsymbol{h}_{1:m}; \boldsymbol{h}^{'}_{1:m}]$ are concatenated as one. The final hidden states (and cell states), denoted $[\boldsymbol{h}_m; \boldsymbol{h}^{'}_m]$, are then passed through a linear MLP layer to reduce the overall dimension. Following, a single LSTM will decode using both the reduced layer output and the embedding vector of the summarisation sequence. Note that the encoder context vector $\boldsymbol{c}^e$ is also involved in the decoder to assist in implementing a decoder attention mechanism (see section 3.2.2).

3.2 Intra-temporal Attention

As explained in Sankaran et al. (2016), an intra-temporal attention is intended to memorise attention focus at previous decoding steps. In standard attention models, LSTMs are responsible for contracting the data but such contraction is done indirectly. By directly assessing areas where focus has been placed, the model is less likely to attend the same segment in a long sequence. Thus reducing repetition. In our model, this attention is applied on both the encoder and decoder.

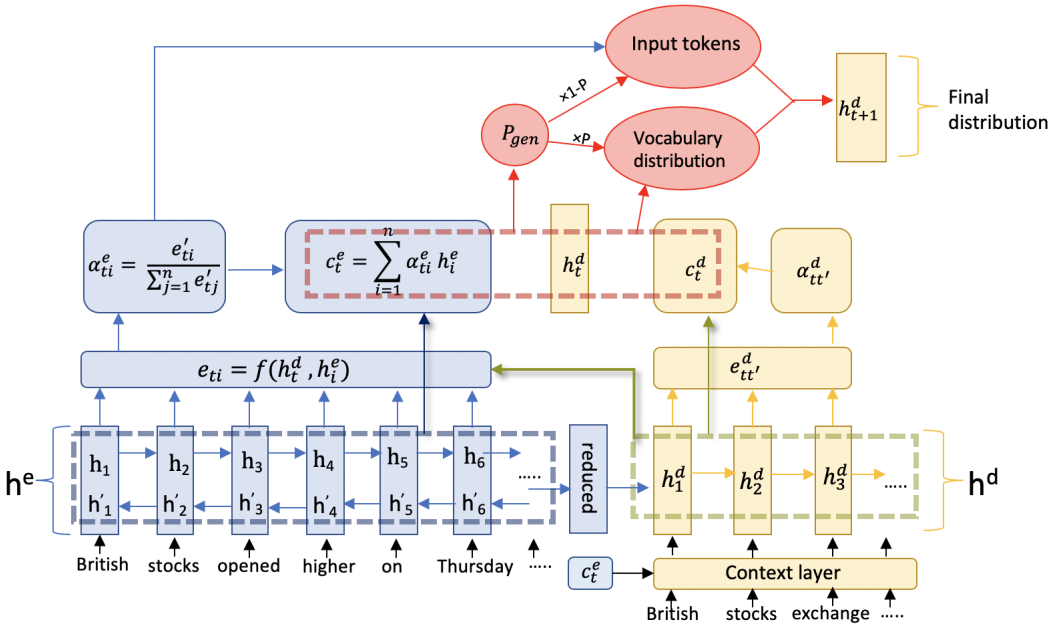

Figure 1: Encoder-decoder attention model with pointer mechanism for abstractive summarisation. e, α and c represent attention score, normalised attention score and context vector respectively. Blue and yellow are attention in encoder-decoder (see sections 3.1 &3.2); Red is pointer mechanism (see section 3.4)

3.2.1 Encoder Attention:

At decoding time step t, the attention score e_{ti} of hidden state $h_i, 1 \leq i \leq m$ can be defined in multiple ways according to Luong et al. (2015):

$$\boldsymbol{e}_{ti} = \begin{cases} (\boldsymbol{h}_t^e)^T \boldsymbol{h}_i^d & \texttt{dot} \\ (\boldsymbol{h}_t^e)^T \boldsymbol{W} \boldsymbol{h}_i^d & \texttt{general} \\ \boldsymbol{v}^T tanh(\boldsymbol{W}[\boldsymbol{h}_i^d; \boldsymbol{h}_t^e]) & \texttt{concat} \\ \ldots & \ldots \end{cases}$$

In our model, we define it as:

$$\boldsymbol{e}_{ti} = f(\boldsymbol{h}_t^d, \boldsymbol{h}_i^e) \quad (1)$$
$$= \boldsymbol{W}_{att}\ tanh(\boldsymbol{W}_e \boldsymbol{h}_t^e + \boldsymbol{W}_d \boldsymbol{h}_i^d + \boldsymbol{b}_{att}) \quad (2)$$

following the same setup in See et al. (2017). Next, to penalise high score of hidden state h_i^e in all previous decoding time steps, we normalise:

$$\hat{\boldsymbol{e}}_{ti} = \begin{cases} exp(\boldsymbol{e}_{ti}) & if\ t = 1 \\ \frac{exp(\boldsymbol{e}_{ti})}{\sum_{j=1}^{t-1} exp(\boldsymbol{e}_{ji})} & otherwise \end{cases}$$

This obtains the penalised source representation at time t, and $\hat{\boldsymbol{e}}_{ti}$ can also be seen as an *alignment model* proposed by Bahdanau et al. (2014) which measures how well the input token i matches with the output token at time t. From which we can compute the context vector through weighted sum:

$$\boldsymbol{\alpha}_{ti}^e = \frac{\hat{\boldsymbol{e}}_{ti}}{\sum_{j=1}^{n} \boldsymbol{e}_{tj}} \quad (3)$$

$$\boldsymbol{c}_t^e = \sum_{i=1}^{n} \boldsymbol{\alpha}_{ti}^e \boldsymbol{h}_i^e \quad (4)$$

In this form, the context vector $\boldsymbol{c}_t^e$ behaves as the expected encoder hidden state representation at decoding time t. This gives h_t^d direct access to the input information so that the decoder relies less on reduced layer output. For this reason, we incorporate the context vector in the decoder by feeding it along with embedding vector of y to a context layer (as shown in Figure 1).

3.2.2 Decoder Attention

To further reduce repetition in long-term sequence prediction, Paulus et al. (2017) also implements intra-attention on decoder, attending previous decoding step. At decoding time step t, a new decoder context vector c_t^d is computed as:

$$\boldsymbol{e}_{tt'}^d = \boldsymbol{W}_{att}^d\ tanh(\boldsymbol{W}_t \boldsymbol{h}_t^d + \boldsymbol{W}_{t'+\boldsymbol{b}_{att}^d} \boldsymbol{h}_{t'}^d) \quad (5)$$

$$\boldsymbol{\alpha}_{tt'}^d = \frac{exp(\boldsymbol{e}_{tt'}^d)}{\sum_{j=1}^{t-1} exp(\boldsymbol{e}_{tj}^d)} \quad (6)$$

$$\boldsymbol{c}_t^d = \sum_{j=1}^{t-1} \boldsymbol{\alpha}_{tj}^d h_j^d \quad (7)$$

3.3 Local Encoder Attention

During the experiment it appears that global encoder attention is computational expensive especially on long articles such as WikiHow. Therefore, we propose a local attention model on only the WikiHow dataset.

We follow the *predictive alignment* as stated in Luong et al. (2015) but with the same function f as our global attention model. The predicted aligned position is:

$$p_t = S \cdot \sigma(\boldsymbol{v}_p^T tanh(\boldsymbol{W}_p \boldsymbol{h}_t^e))$$

where S is the input article length. The context vector $\boldsymbol{c}_t^e$ is then defined as a weighted sum of hidden state $\boldsymbol{h}_i^e$ within the window $[p_t - D, p_t + D]$. To favour the tokens near p_t, a Gaussian distribution centered at p_t is attached to the weight $\boldsymbol{\alpha}_{ti}^e$:

$$\boldsymbol{\alpha}_{Gauss,ti}^e = \boldsymbol{\alpha}_{ti}^e \; exp\big(-\frac{(i-p_t)^2}{2\sigma^2}\big)$$

3.4 Pointer Mechanism

In the token generation stage, the model follows the "Switching Generator-Pointer" proposed in Nallapati et al. (2016) to select between– **A**. generating tokens from vocabulary; **B**. a pointer to one of the token position in the source. The switch uses a sigmoid activation on the sequence context, denoted as:

$$P_{gen} = \sigma(\boldsymbol{W}_{gen}[\boldsymbol{c_t^e}; \boldsymbol{h_t^d}; \boldsymbol{c_t^d}] + \boldsymbol{b}_{gen})$$

`(generate from vocab)`

$$P_{point} = 1 - P_{gen}$$

`(generate from input tokens)`

The main difference between the two types of generation is in the pointer mechanism (**A**) being able to generate out-of-vocabulary words. The probability distribution for the vocabulary is computed:

$$\begin{aligned} P_{vocab} &= p(\hat{\boldsymbol{y}}_t | P_{gen}) \\ &= softmax(\boldsymbol{W}_{vocab}[\boldsymbol{c}_t^e; \boldsymbol{h}_t^d; \boldsymbol{c}_t^d] + \boldsymbol{b}_{vocab}) \end{aligned}$$

over all the words in vocabulary integrated with contextual information.

The probability distribution of input tokens is computed using temporal attention weights:

$$P_{input} = p(\hat{\boldsymbol{y}}_t | P_{point}) = \boldsymbol{\alpha}_{ti}^e$$

The final probability distribution of predictive tokens is:

$$P(\hat{\boldsymbol{y}}_t) = P_B P_{vocab} + (1 - P_B) P_{input} \quad (8)$$

4 Model Objectives

4.1 Baseline

The training objective is a weighted sum of cross-entropy loss and self-critical ROUGE reward.

$$L_{rl} = (r(\hat{\boldsymbol{y}}) - r(\boldsymbol{y}^s)) \sum_{t=1}^{n} log\; p(\boldsymbol{y}_t^s | \boldsymbol{y}_1^s, ..., \boldsymbol{y}_{t-1}^s, \boldsymbol{x}) \quad (9)$$

$$L_{ml} = -\sum_{t=1}^{n} log\; p(\boldsymbol{y}_t^* | \boldsymbol{y}_1^*, \boldsymbol{y}_2^*, \ldots, \boldsymbol{y}_{t-1}^*, \boldsymbol{x})$$

$$L_{hybrid} = \gamma L_{rl} + (1 - \gamma) L_{ml} \quad (10)$$

where $\{\boldsymbol{y}_1^*, \boldsymbol{y}_2^*, \ldots, \boldsymbol{y}_n^*\}$ is the ground-truth output; $\{\boldsymbol{y}_1^s, \ldots, \boldsymbol{y}_n^s\}$ is the model's output sequence; and $\hat{\boldsymbol{y}}$ is the baseline output obtained by maximising the output distribution at each time step, such that the probability of outputs with smaller rewards value than the baseline is minimised and reward higher is maximised. The reward function $r(\boldsymbol{y})$ can be any evaluation metric comparing with ground-truth $\boldsymbol{y}^*$. In our baseline, function $r(\boldsymbol{y})$ is set as the F score of ROUGE-L, following the setup in Paulus et al. (2017).

4.2 Variation by Training Objective

The baseline had a hybrid training objective of maximum likelihood (ML) loss L_{ml} and RL-based reward function. This paper offers 2 other comparisons of the "extreme" objectives. Pure ML and RL models by setting $\gamma = 0$ and 1.

4.3 Variation by Reward

DSR functions (such as BERTScore Zhang et al. (2019)) are alternatives for $r(\boldsymbol{y})$ which focus on sentence-level generation evaluation. By exploiting semantic relationships, the outputs of text generation models give relatively higher correspondence with human evaluation. We replace $r_{ROUGE}(\boldsymbol{y})$ in equation (9) with $r_{BERTScore}(\boldsymbol{y})$.

$$L_{rl(B)} = (r_B(\hat{\boldsymbol{y}}) - r_B(\boldsymbol{y}^s)) \sum_{t=1}^{n} log\; p$$

For a full suite, there are 4 model variants:

1. Maximum Likelihood (ML)
2. Reinforcement Learning (RL) with ROUGE-L Reward (RL(R))
3. RL with BERTScore Reward (RL(B))

4. Hybrid (ML + RL(R)) Objective (hybrid)

The essence of RL models is to maximise reward functions directly. Thus the initial hypothesis is for the pure RL models to outperform the other models when evaluated on their respective reward functions.

Examining the BERTScore vs. ROUGE, the former is the "softer" metric. BERTScore compares vector representations of texts and scores them for similarity. It is therefore a comparison of the meanings attached to sentences. For tokenised reference sentence $y^* = \langle y_1^*, \ldots, y_k^* \rangle$, map y^* into a sequence of vectors $\langle \mathbf{y_1^*}, \ldots, \mathbf{y_k^*} \rangle$. Similarly map tokenised output sentence $y^s = \langle y_1^s, \ldots, y_k^s \rangle$ as $\langle \mathbf{y_1^s}, \ldots, \mathbf{y_k^s} \rangle$. The following are the recall, precision and F scores for BERT

$$R_{BERT} = \frac{1}{|y^*|} \sum_{y_i^* \in y^*} \max_{y_j^s \in y^s} \mathbf{y_i^{*T}} \mathbf{y_j^s}$$

$$P_{BERT} = \frac{1}{|y^s|} \sum_{y_i^s \in y^s} \max_{y_j^* \in y^*} \mathbf{y_i^{*T}} \mathbf{y_j^s}$$

$$F_{BERT} = 2\frac{P_{BERT} \cdot R_{BERT}}{P_{BERT} + R_{BERT}}$$

ROUGE scores work based on exact-string matches, which is a blunt method of comparison. There are 3 types of ROUGE scores used in this paper's model evaluation:

1. **ROUGE-1** unigram overlap
2. **ROUGE-2** bigram overlap
3. **ROUGE-L** longest common subsequence

By the metrics' characteristics, we would expect that models trained on ROUGE would not have poor BERTScores; whereas models trained on BERTScore might not have good ROUGE scores.

5 Datasets

In this paper, model variants were applied on 3 datasets: 1. the standard **CNN/DailyMail** dataset (**287K**) (Hermann et al., 2015); 2. **Gigaword** (**3.8M**) (Graff et al., 2003); 3. **WikiHow** (**161K**) (Koupaee and Wang, 2018). The first 2 are common datasets used for summarisation. The distribution of article lengths for the various datasets are plotted on Figures 2, 3 and 4.*

*The y axis is the is the frequency in 5k-size random samples and CNN/DM and WikiHow have the same bin size.

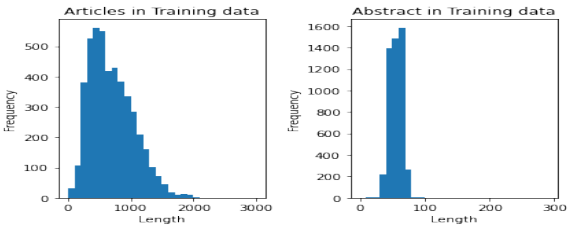

Figure 2: CNN/DailyMail Training

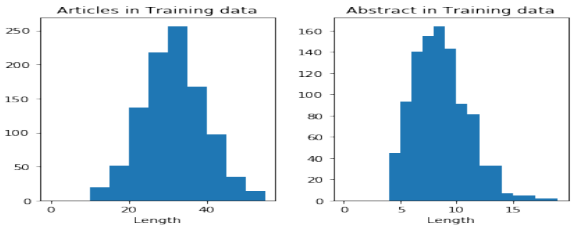

Figure 3: Gigaword Training

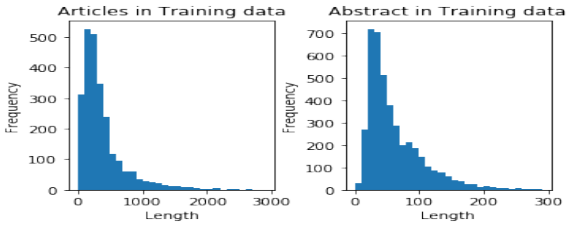

Figure 4: WikiHow Training

For the CNN/DailyMail (CNN/DM) dataset (see Fig 2), though there are article lengths of up to 2000, abstract lengths only go up to 100. The distribution of the abstract length is also less skewed compared to article lengths. This might indicate that abstracts in this dataset prioritize compactness. Whereas for the WikiHow dataset (see Fig 4), abstract lengths range to 400 and the distribution shape fits more proportionally with the original passage length.

Now reviewing article contents. The CNN/DM dataset consists mainly news articles and short summaries. Gigaword dataset is a general set of short passages and condensed versions of these passages. Passages in the WikiHow dataset are instructional articles. These passages are authored by the general public and hence less likely to conform to a single style of writing. The WikiHow dataset in this light is more general than CNN/DM, and also harder to analyse.

6 Experiments and Results

In the hybrid model, we fix $\gamma = 0.5$ across all datasets. This is different from the setting in

Paulus et al. (2017)'s setting of $\gamma = 0.9984$. The full hyper-parameter setting is shown in table 6 (Appendix B).

6.1 Training

We first establish a pre-training(ML) for 10,000 iterations with a learning rate of 0.001. Note that in practice, to reduce the effect of exposure bias, we pass the previous generated tokens as input with probability 0.25 instead of ground truth label. In later ML training, learning rate is kept at 0.001 and for RL training, learning rate is set at 0.0001.

In the Gigaword dataset, the pure ML, hybrid, and RL(R) models were trained for a further 10,000 iterations from the pre-training while the RL(B) model was trained for a further 4,500 iterations from the pre-training.

In the CNN/DM and WikiHow dataset, as BERTScore computational cost is expensive, the RL(B) was trained for a further 3,000 iterations.

Due to the larger size of the WikiHow dataset and thus computational limitations, the pure ML, hybrid, and RL(R) models were trained for a further 6,000 iterations from the pre-training while the RL(B) model was trained for a further 5,100 iterations from the pre-training.

6.2 Evaluation and Testing

The models are then validated on the basis of the earlier mentioned ROUGE scores– 1. ROUGE-1, ROUGE-2, and ROUGE-L; and BERTScore. The ROUGE-1 unigram is indicative of similarity in content. However, because terms are compared individually, it cannot identify text coherence. ROUGE-2 bigram does encompass ties between neighbouring terms and is a step up in sensing adjacent word relationship. ROUGE-L measures the Longest Common Subsequence encourages text generation that is fluid and at the same time captures information. BERTScore as previously mentioned is much more information based than ROUGE scores. A high BERTScore suggests good overlap in content between output and reference texts.

The following tables show the ROUGE and BERTScores (on the test set*) for the 3 datasets:

Interestingly, the hybrid model did not consistently outperform the ML model despite having partial RL features. In fact, it is only in the Gigaword dataset that the Hybrid model outperforms the pure ML model. It is possible that by constructing a composite objective, the model does not juggle well between maximising the individual objective components. It cannot exploit the advantages that RL and ML models offer independently.

*Models with highest score during validation are selected to perform testing. Also note that the ROUGE-1,2,l scores testing are using the same model.

Table 1: CNN/DM Dataset Test Scores

Model	Rouge1	Rouge2	RougeL	BERT
ML	0.4365	0.2059	0.4002	0.8967
Hybrid	0.4317	0.2041	0.3986	0.8960
RL(R)	0.4062	0.1929	**0.4369**	0.8839
RL(B)	**0.4502**	**0.2166**	0.4089	**0.9021**

Table 2: Gigaword Dataset Test Scores

Model	Rouge1	Rouge2	RougeL	BERT
ML	0.4212	0.2205	0.4063	0.9030
Hybrid	**0.4233**	**0.2223**	0.4081	0.9032
RL(R)	0.3291	0.1591	**0.4326**	0.8767
RL(B)	0.4204	0.2092	0.4034	**0.9045**

Table 3: WikiHow Dataset Test Scores

Model	Rouge1	Rouge2	RougeL	BERT
ML	**0.3433**	**0.1466**	0.2558	0.8752
Hybrid	0.3170	0.1337	0.2516	0.8655
RL(R)	0.2662	0.0942	**0.3018**	0.8831
RL(B)	0.3377	0.1435	0.2438	**0.8987**

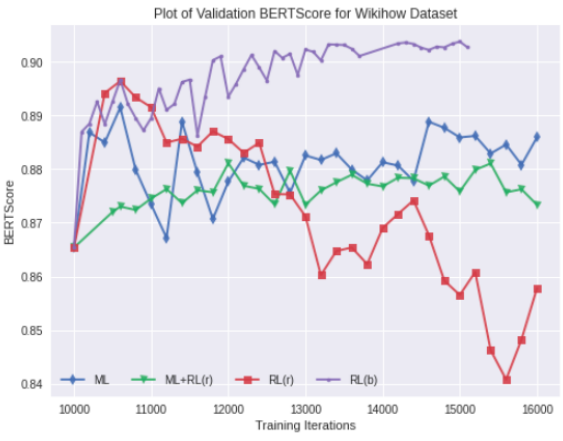

Figure 5: WikiHow Validation BERTScores against Iterations

As expected, for all 3 datasets, the model using RL with reward function corresponding to ROUGE-L scores performed best on ROUGE-L. But as we see in Tables 1 and 2, despite having the

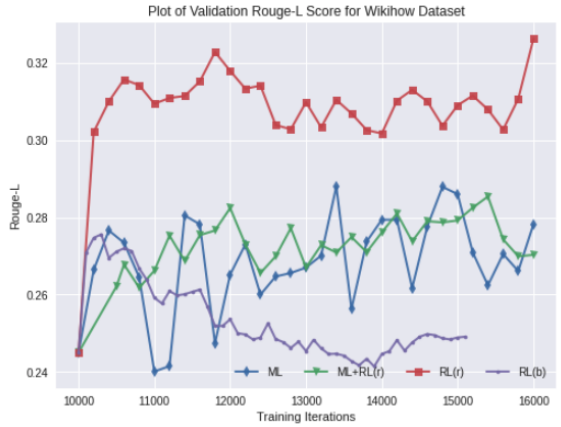

Figure 6: WikiHow Validation ROUGE-L against Iterations

highest ROUGE-L score, the RL(R) model fared the worst on every other metric. Scores for the other models are comparable with each other.

Referring to Fig 5* which is the plot of validation scores, RL(R) is the only model observed to have a decreasing trend on BERTScores through the iterations. Contrasted with the performance of RL(B) in Fig 6 as the only model with a decreasing trend on ROUGE-L. In general, the improvements in performance of ML and hybrid models appear more consistent than the pure RL models.

Qualitatively, from a general inspection of the text output from the models, there are certain peculiarities observed. Summaries generated by the RL(R) models tend to feature more repetition and are not as readable compared to summaries by other models. RL(B) summaries are more detailed. It is not uncommon to find RL(B) summaries containing more information than reference summaries.

The hybrid model is a combination of the pure ML and RL(R) model. Yet, the summaries generated by the hybrid model are more similar to ML generated summaries than RL(R). It is interesting to see the dominance of the ML component with $\gamma = 0.5$.

Figs 7, 12, 13 (Appendix B) plot the ROUGE-L and BERTScores for the validation runs of the datasets. There is an apparent inverse relationship between ROUGE-L scores and BERTScores. Quite similarly to the earlier plots in Fig 5 and 6, the hybrid model is the most stable and scatter points are most concentrated. There is a large variation in the RL models especially for RL(R).

*More validation plot on other two datasets can be found in Appendix B

Table 4: Gigaword Sample Summaries

Article	**an american woman running for a seat on the athens city council hopes to become one of the first-ever immigrants to hold public office in a country struggling to assimilate hundreds of thousands of foreigners who have decided to call greece home .**
Ref	**first black american candidate joins immigrants seeking office**
ML	american woman hopes to become first-ever immigrants to hold public office
Hybrid	american woman hopes to become first-ever immigrants to hold public office in greece
RL(R)	american woman running for immigrants to greece immigrants to greece home to greece immigrants to
RL(B)	american woman running for immigrants to hold public office in greece

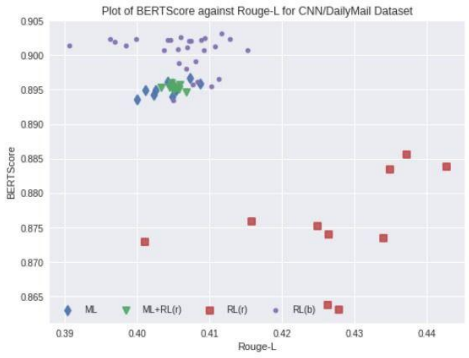

Figure 7: CNN/DM BERTScore against ROUGE-L

6.3 Human Evaluation

To further investigate metrics ROUGE and BERTScore, we also carried out human evaluation on 20 unlabelled mixed testing examples generated by $RL(R)$ and $RL(B)$ models for each of the three datasets. Mean scores are shown in Table 5.

Table 5: Human Evaluation on Readability, Relevance and Overall Performance

	Giga		**CNN/DM**		**Wiki**	
	R_r	R_b	R_r	R_b	R_r	R_b
Read	5.45	**7.85**	5.20	**7.45**	2.00	**2.55**
Rel	6.85	**7.70**	6.30	**7.30**	**1.80**	1.65
Ovrl	6.05	**7.80**	5.85	**7.35**	1.85	**2.05**

6.4 Local Attention Refinement

We also implement a local attention model as stated in Section 3.3 for RL(B) objective function with window sizes $D = 50, 100$. The validation scores are plotted in Fig 14. Testing BERTScores are 0.89274 and 0.89615 respectively. With local attention, the model learns faster but is more uncertain. This is mediated by larger window size. Test examples can be found in Table 8. Future works can consider exploration on local attention.

7 Discussion

Comparing the overall scores across the 3 datasets, scores are much lower on the WikiHow dataset than the other 2. The writing style in WikiHow dataset is much more diverse since its contributors are general public. The passages are thus less refined. This arguably makes it a more difficult dataset to be trained on; since models have to be trained to evaluate over a range of writing styles.

A notable observation from the results is the compromise in ROUGE-1, ROUGE-2 and BERTScores that RL(R) models have while trying to maximise ROUGE-L reward. It is important to consider what this phenomenon means for models meant to optimise ROUGE-L scores. It may not be worthwhile to scrimp on information. In conjunction with the observation that despite the purpose of ROUGE-L favoring in-sequence matching, RL(B) models generate more fluid text. It is possible that by training on a more restrictive or strict metric like ROUGE-L could be akin to teaching models particular phrases but that may not be congruous for general texts.

In terms of model training, pure ML and hybrid models appear to have improving ROUGE-L and BERTScores compared to the pure RL models. But between the ML and hybrid models, ML shows much more fluctuation although an increasing trend can still be traced. Given the differing performance of RL(R) and RL(B) models, it demonstrates the challenge of selecting appropriate reward functions for RL models. It is possible that the nature ML models provide general improvement direction despite its instability.

The repetitive characteristic of the RL(R) could be a result of the model's efforts to generate as long a "good subsequence" of words. The RL(R) summaries are longer due to the repetition and is generating words that do not add informational value. This is an unwelcome outcome of the model focusing on generating a single long phrase. This shifts the model's attention on a single piece of information and trying to generate a good paraphrase of that information.

For the scatter plots of BERTScores against ROUGE-L, the variance within the models is greater for the harder datasets like in Fig 13. But with easier datasets like Gigaword in Fig 12, it is more obviously identifiable the disparity between models since clusters are more concentrated within themselves. Across plots, there is an obvious divide between the RL(R) models and even within RL(R) there is a great variation in scores.

These observations suggest that using BERTScore as an alternative reward function is not a bad choice. Despite slight dips in ROUGE-L scores, RL(B) models can still produce readable summaries with good informational content. Further, lower ROUGE-L scores are not entirely indicative of bad readability nor content; just a different composition of information.

8 Conclusion and Future Works

In this paper, we examine 4 different models with 2 different styles of metrics. ROUGE scores require answers that are more precise. Thus focusing on specific wordings, especially ROUGE scores that compare multiple matchings. Whereas BERTScore is focused on meaning. In comparing the performance of each model on the 2, we are able to analyse the characteristics of the models. Scores from ROUGE-1 and BERTScore generally agree and help to support judgements on the strength of content extraction that models provide.

The quantitative and qualitative analysis of the summaries demonstrate that these metrics can only represent the quality of summaries to a limited extent. At the end, qualitative analysis is still necessary to examine the summaries. The downsides of using ROUGE-L as both a metric and a reward function is also exposed when placed against BERTScore. This is also in conjunction with the broad observation of an inverse relationship between BERTScores and ROUGE-L scores.

With respect to the hybrid model, while the numbers for ML are better, the plots do show hybrid models exhibiting greater stability. In view of Paulus et al. (2017)'s paper that experimented with a lower composition of ML in the hybrid model, it might be worth varying γ to observe the changes in the hybrid model performance for future works.

References

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. *ICLR 2017*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Regina Barzilay, Kathleen R. McKeown, and Michael Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, page 550–557.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *arXiv preprint arXiv:1805.11080*.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*.

CK Jackie Cheung. 2008. Comparing abstractive and extractive summarization of evaluative text: controversiality and content selection. *B. Sc.(Hons.) Thesis in the Department of Computer Science of the Faculty of Science, University of British Columbia*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *arXiv preprint arXiv:1506.03340*.

Hongyan Jing and Kathleen R. McKeown. 2000. Cut and paste based text summarization. In *n Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, pages 178–185.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Siyao Li, Deren Lei, Pengda Qin, and William Yang Wang. 2019. Deep reinforcement learning with distributional semantic rewards for abstractive summarization. *arXiv preprint arXiv:1909.00141*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2015. Sequence level training with recurrent neural networks. *ICLR 2016*.

Baskaran Sankaran, Haitao Mi, Yaser Al-Onaizan, and Abe Ittycheriah. 2016. Temporal attention model for neural machine translation. *arXiv preprint arXiv:1608.02927*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Hyperparameter and implementation details

Table 6: Hyper-parameter setting

	CNN	Giga	Wiki
hidden dim	400	512	512
embed dim	200	256	256
batch size	50	200	10
max enc length	400	55	500
max dec length	100	15	150
beam size	5	4	4
vocab size	30000	50000	20000

In Paulus et al. (2017) paper, they used $\gamma = 0.9984$ for the ML+RL loss function, 200 dimension encoder bi-LSTM hidden dimension and 400 decoder hidden dimension. The vocabulary size is 150,000. The embedding dimension is 100 and the batch size is 50. Also they used beam search of width 5.

B Appendices

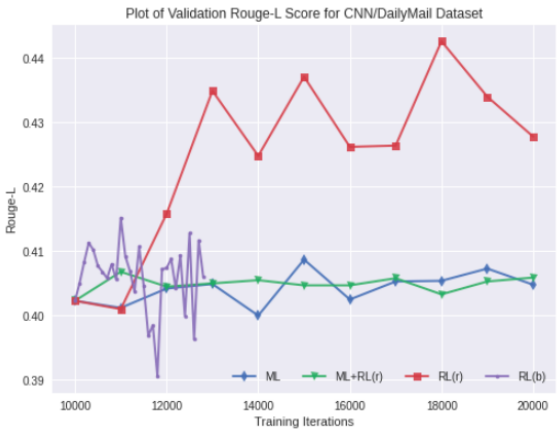

Figure 8: CNN/DM Validation ROUGE-L against Iterations

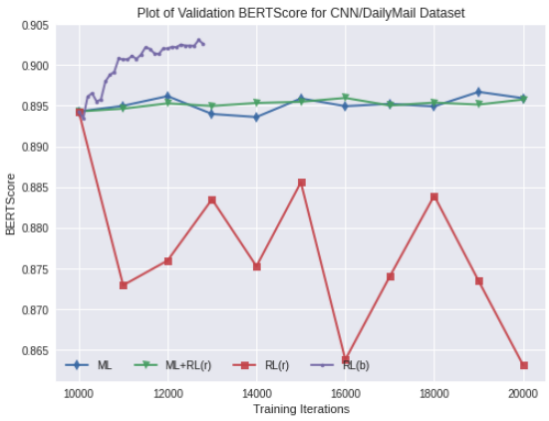

Figure 9: CNN/DM Validation BERTScore against Iterations

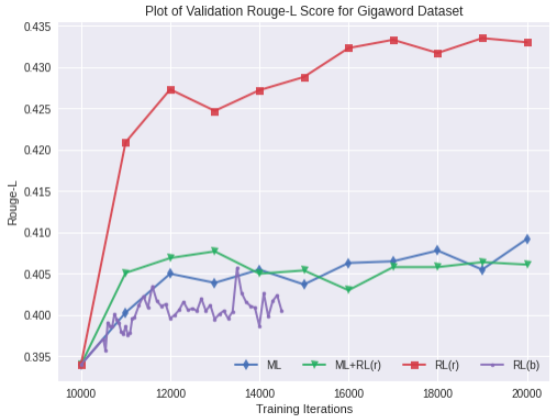

Figure 10: Gigaword Validation ROUGE-L against Iterations

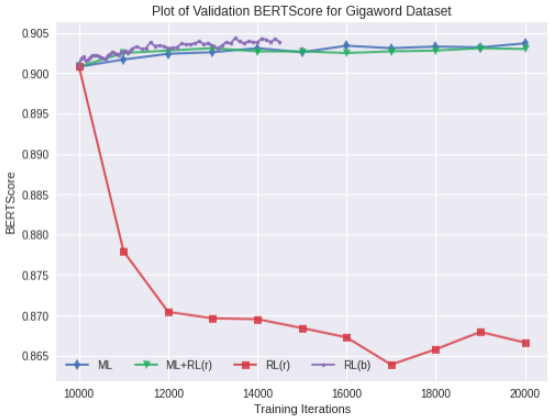

Figure 11: Gigaword Validation BERTScore against Iterations

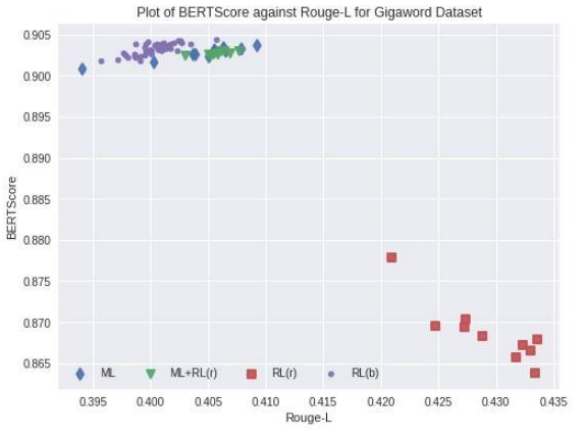

Figure 12: Gigaword BERTScore against ROUGE-L during validation

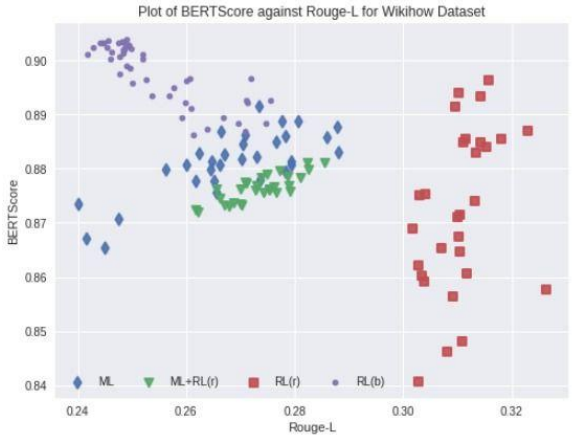

Figure 13: WikHow BERTScore against ROUGE-L during validation

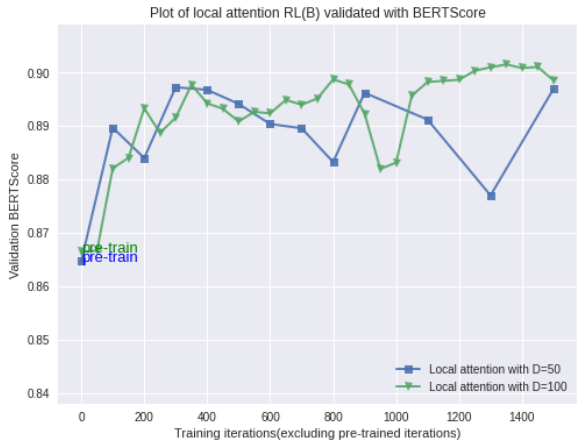

Figure 14: WikiHow BERTScore validation using local attention model

C Supplemental Material(Examples printout)

Table 7: CNN/DailyMail Sample Summaries

Article	**a stern defensive line has twice helped jose mourinho and chelsea to premier league glory , with a third likely to follow upon the resolution of the 2014-15 campaign . statistics show the chelsea manager is the best in premier league history at shutting out opponents , edging out the man he replaced upon returning to london for his second stint at chelsea , rafa benitez . of the 189 premier league matches the portuguese has overseen , he has an astounding 101 clean sheets , nullifying his rivals in 53.4 per cent of games . chelsea boss jose mourinho has the best clean sheets percentage record in premier league history . a stubborn defence has twice helped mourinho win the league , with a third likely at the end of the season . mourinho -lrb- left -rrb- beats rafa benitez in the clean sheets record and average of goals conceded . benitez has 13 more , 114 from 254 matches with liverpool and chelsea , keeping out his opponents in 44.9 per cent of his games . from his 133 games at manchester city , roberto mancini edged out sir alex ferguson by 0.1 per cent . ferguson retired at old trafford with 359 shut-outs and clean sheet percentage of 44.3 , but mancini had 44.4 per cent with 59 from 133 . mourinho also heads the list of managers with the lowest average of goals conceded per game , with 0.63 . his chelsea teams have collectively only conceded a stubborn 120 times under his guidance . again , benitez is second but with a much bigger difference , as the spaniard averages 0.82 goals every premier league encounter . mancini is third , on 0.83 and bruce rioch fourth on 0.84 , but the former arsenal manager has taken charge of the lowest number of games -lrb- 38 -rrb- in the top 10 . carlo ancelotti is fifth , averaging 0.86 goals a game from the two seasons he oversaw at chelsea , where he also won the premier league . sir alex ferguson -lrb- right -rrb- celebrates winning the league with one of his best defenders , rio ferdinand . arsene wenger has also established an imposing defensive record from 708 games at arsenal . roberto mancini -lrb- left -rrb- and carlo ancelotti have impressive defensive records from shorts spells in england . ferguson , a 13-time league champion from his time at united , conceded 703 times and averaged 0.87 goals per game . arsene wenger , the second-longest serving manager in premier league history with 708 games to date , has kept 277 clean sheets and conceded 672 goals . he ranks ninth in the list of clean-sheet percentages and eighth in the number of goals conceded per game , averaging 0.95 .**
Ref	**<t> jose mourinho has a stunning defensive record from his time at chelsea . </t> <t> he has 101 clean sheets from 189 games , conceding only 120 goals . </t> <t> his record surpasses those of rafa benitez and sir alex ferguson . </t> <t> mourinho : i have a problem , i am getting better and better . </t> <t> click here for all the latest chelsea news . </t>**
ML	<t> chelsea boss jose mourinho has the best clean sheets percentage record in premier league history . </t> <t> benitez has 13 more , 114 from 254 matches with liverpool and chelsea , keeping out his opponents in 44.9 per cent of his games . </t> <t> carlo ancelotti is fifth , averaging 0.86 goals a game from the two seasons he oversaw at chelsea , where he also won the premier league . </t> <t> mourinho also heads the list of managers with the lowest average of goals conceded per game , with 0.63 . </t>
RL(R)	<t> stern defensive line has twice helped jose mourinho and chelsea to chelsea . rafa benitez . </t> <t> chelsea manager is the best in premier league history at chelsea . rafa benitez . </t> <t> mourinho has the clean sheets percentage record in premier league . </t> <t> mourinho has the 101 clean sheets . </t> <t> mourinho has the 101 clean sheets , rafa benitez . rafa benitez in the clean sheets record and average of goals . </t> <t> mourinho has the 101 clean sheets at chelsea . </t> <t> mourinho has the best in premier league
RL(B)	<t> stern defensive line has twice helped jose mourinho and chelsea to premier league glory . </t> <t> chelsea manager is the best in premier league history at shutting out opponents . </t> <t> chelsea boss jose mourinho has the best clean sheets percentage record in premier league history . </t> <t> benitez has 13 more , 114 from 254 matches with liverpool and chelsea . </t>

Table 8: WikiHow Sample Summaries

Article	**Fix their relative height and distance so that you can sit comfortably. Don't set your seat so far back that you have to lean forward to grip the steering wheel. Avoid placing undue stress on your body, which may tire you out and distract you, making you less responsive.The positioning of your seat may effect which grip you find more comfortable: 9-and-3 or 10-and-2. Taller people, for instance, may find 10-and-2 most comfortable, due to the limits of how much they can adjust either the steering column or their seat. Extend your sights at least a half-mile to a mile farther up the road. Keep your eyes peeled for any curves, hazards, or other factors that may necessitate a change in direction. Anticipate when you need to turn early on. Give yourself as much time as possible to plan and execute changes in direction.If you are passing through a tight curve that significantly reduces your field of vision, always focus on the farthest point that you can see ahead of you. Trust your peripheral vision to alert you to sudden changes that appear closer to hand. Anticipate that a change in direction at slow speed will require greater physical effort with the steering wheel. Be prepared to turn it by a greater number of degrees in low-speed areas like parking lots, residential streets, and urban neighborhoods. Conversely, keep your turning actions with the wheel very, very slight when driving fast. Expect a slight turn of the wheel to cause a very pronounced change of direction on high-speed roads like highways., Turning the steering wheel when the car is parked or otherwise at rest can have adverse effects on your tires and power-steering. Do so when necessary, such as when you parallel-park or execute a K-turn. Otherwise, try to avoid it., Maintain optimum control over the vehicle while using tools other than the steering wheel. Use your nearest hand to operate such functions as turn signals and gear shifts while driving. Keep your other hand where it is as you do so. Don't risk letting go of the wheel to alter its position.**
Ref	**<t> adjust your seat and steering column properly. </t> <t> look farther down the road. </t> <t> factor in your speed when steering. </t> <t> keep "dry steering" to a minimum. </t> <t> practice safe one-handed steering. </t>**
ML	<t> set your seat </t> <t> focus on the [UNK] </t> <t> focus on the steering wheel. </t> <t> change the steering wheel. </t>
Hybrid	<t> sit comfortably. </t> <t> focus on the [UNK] </t> <t> avoid [UNK] </t> <t> avoid [UNK] </t>
RL(R)	<t> sit your seat of to sit comfortably. </t> <t> turn on your wheel in of in in in of you. </t> <t> turn on the wheel in of on in on on on on on on on on on on on on on on on on on on on on on on on on on on on on of to to of to alter of to the wheel to alter of to the wheel to alter of to alter of to alter of to alter of to of to alter of to the wheel to alter of to alter of to of
RL(B)	<t> set your seat wheel. </t> <t> practice your seat regularly. </t> <t> practice the position. </t>
RL_{loc}^{D50}	<t> set your seat </t> </t> <t> assess your peripheral position. </t> <t> practice your steering wheel. </t>
RL_{loc}^{D100}	<t> set your seat wheel. </t> <t> plan your seat </t> <t> turn your turning </t> <t> keep your turning </t>