Exploring Generalisation in Deep Learning Abstractive Summarisation Models

Student Name: [redacted]

Supervisor Name: [redacted]

Submitted as part of the degree of M.Sc. MISCADA to the

Board of Examiners in the Department of Computer Science, Durham University.

Abstract **–** Deep Learning has been used to build abstractive summarisation models. While the outcome of these models is impressive, the generalisation behaviour is not clearly explored. It's uncertain if the models will have the same performance when evaluated with out-of-distribution data. In this project, we analysed the generalisation ability of two famous architectures, RNN sequence-to-sequence and Transformer models. We diagnosed the models' behaviour based on different aspects and evaluated their generalisation ability with several measures. We conducted an in-depth analysis to understand the limitation of the selected models by experiment with different datasets' characteristics and dataset bias. We show that the Transformer is more robust to out-of-distribution datasets. The analysis of the Transformer model with out-of-distribution data implies that the model has generalisation ability with promising performance. Finally, we show directions for future work to improve the generalisation ability of the model.

Keywords – Abstractive Summarisation, Generalisation, Transformer, RNN sequence-to-sequence, cross-dataset, dataset bias.

I. INTRODUCTION

As the amount of documents and data keeps increasing, the demand for automatic text summarisation that produces short coherent summaries increases. Text summarisation is the process of generating short, meaningful summaries for input documents or long sentences. Summarising approaches differ based on different aspects: the length of the input document (if it is a single page or multiple pages); its purpose (query-based, domain-specific, or general); and the summarisation strategy. There are two types of summarisation strategies: extractive and abstractive. Extractive summarisation (Moratanch and Chitrakala, 2017) is focusing on extracting key sentences from long documents as a summary, while abstractive (See et al., 2017) is generating sentences that capture the salient ideas of the documents. These sentences do not necessarily exist in the input.

Most of the state of the art methods for solving abstractive text summarisation use deep neural network involving recurrent neural network (RNN) sequence-to-sequence model that follows encoder-decoder architecture (Cho et al., 2014). Many researchers extend this framework by adding more mechanisms for handling different problems like repetition or handling rare data. Recently, the Transformer (Vaswani et al., 2017), a new model that redesigns the encoder-decoder architecture without involving RNN on both encoder and decoder, has been adopted. RNNs are used in the sequence-to-sequence problem to handle the sequencing and ordering of data. Whereas, the encoder and decoder in the Transformer apply positional encoding to capture the ordering of the input. Moreover, The Transformer is faster to train since it allows for more parallelisation.

Recent research has focused on building abstractive summarisation models with different architectures published rapidly. However, the generalisation ability of these models to handle unseen and new data distributions is not clearly explained. Abstractive summarisation models have complex architectures that require days for training, and it is essential to address the generalisation ability of these models with different datasets. Evaluation in prior research is done using the same dataset (in-dataset), which means that train and test data have the same distribution and are derived from the same dataset. However, it is crucial to understand how these models will behave with out-of-distributions datasets (cross-dataset) since, in real usage, it will be inefficient and costly to train our model on all possible new distributions, and since we are dealing with text, models will always encounter new and unexpected distributions.

In this project, we are going to diagnose generalisation ability to out-of-distribution datasets for the two most famous architectures in abstractive summarisation problem – RNN-sequence-to-sequence and Transformer.

A. Objectives

The proposed research question is: *To what extent do abstractive summarisation models have generalisation ability?*. We will diagnose the generalisation ability from different angles. First, we will analyse the models' behaviour with cross-datasets, which are the datasets that have a different distribution than the training dataset. Moreover, we will demonstrate the effect of dataset's characteristic into abstractive models output. Finally, we will study the models' ability to handle imbalanced categories in training datasets. To answer the research question, the objectives of the project were divided into different levels: basic, intermediate, and advanced.

The **basic** objective is to establish a candidate sequence-to-sequence architecture model for the abstractive summarisation task and implement the model. The selected architecture should perform well with abstractive summarisation and should address common issues in summarisation by adopting different mechanisms and design choices. The purpose is to have a strong understanding of Deep Learning (DL) abstractive sequence-to-sequence models, to implement a model for the study, and to conduct in-dataset evaluation.

The **intermediate** objective is to implement the Transformer model and conduct in-dataset evaluation, as well as research and analyse different datasets to be selected for cross-dataset evaluation. The project will focus on datasets from the news domain since most available datasets for summarisation problem are from the news domain. A cross-dataset evaluation for both models will be conducted. Additionally, the evaluation will address the models' behaviour based on different characteristics of the cross-datasets, such as coverage and density.

Finally, the **advanced** objective is to study the effect of topic bias on the output of the implemented models. This step will evaluate the models' behaviour on different categories in the dataset for in-dataset and cross-datasets. The purpose of this objective is to study if abstractive models are affected if the training dataset contains under-represented categories.

B. Contributions

Prior research has focused on improving abstractive summarisation models. However, a lot of work is needed to fully understand the generalisation of these models. In this project, we diagnosed the generalisation ability of two famous abstractive summarisation models and analysed the generalisation from multiple perspectives. The project revealed that:

- The Transformer model generates more abstractive summaries while RNN Seq2Seq model summaries are skewed towards extractive;
- The Transformer model is more robust to out-of-distribution datasets;
- There is a need for a new evaluation metric for abstractive summarisation;

- Abstractive models' output characteristics are affected by the training dataset; and
- Abstractive models are not affected by topic bias in the dataset.

II. Related Work

A. *Abstractive summarisation Techniques*

Techniques for Abstractive summarisation are classified into structure-based and semantic-based approaches (Gupta and Gupta, 2018). Structure-based approaches use different methods to select the essential information to create abstractive summaries, while semantic-based approaches create the semantic representations of the text and then use natural language generation to generate abstractive summaries. Recently, deep learning techniques have succeeded in generating abstractive summaries that understand the complex relations within the text and identify both semantic and structural information.

B. *RNN Sequence-To-Sequence Model*

The sequence-to-sequence framework is used in text generation applications such as machine translation (Bahdanau et al. 2014), text summarisation, and speech to text applications (Bahdanau et al. 2016), where the input is a sequence of data and the output is a sequence of text. Deep Neural Network has recently been widely used in building this framework. Cho et al., 2014, has introduced RNN encoder-decoder architecture for machine translation as shown in Figure 1. The encoder is mainly RNN (Recurrent neural network) layer or stack of RNN layers that process the input sequence and return its internal state as input to the decoder layer. The decoder is also a stack of RNN layers that is trained to predict a word at each time-step. RNN is a class of Neural Networks that is suitable for modelling sequential data. An RNN layer contains a cycle that iterates over each element of the sequence and maintains an internal state that depends on the previously seen elements up to a time-step. Moreover, RNNs are useful because of their ability to handle any lengths of the input and output.

The encoder takes an input sequence of words and processes one word at a time-step. It first converts the input into a fixed vector through the embedding layer that will encode it into an internal representation. Then the vector will go into RNN layer as input which will compute the hidden state of the hidden layer. This computation will be repeated at each input of the sequence. The computation of the hidden state takes the previous hidden state and the new input data. The state of the last hidden layer is called the context vector, and it is affected by all previous states. The context vector is fed into the decoder as its initial state. Then we use a start-of-sequence as input to the decoder. The decoder will compute its internal hidden state at each step using the previous state and actual output. Then an activation function (e.g. softmax) is used to produce probabilities that are used to generate the predicted output. At the end, an end-of-sequence is generated or maximum length is reached. After that, we compare the actual and predicted outputs of the sequence and compute the loss. The error is back-propagated into previous hidden layers in time-steps to update the weight of the parameters. During training, the decoder uses the teacher forcing technique (Goldberg, 2017:196), where the input

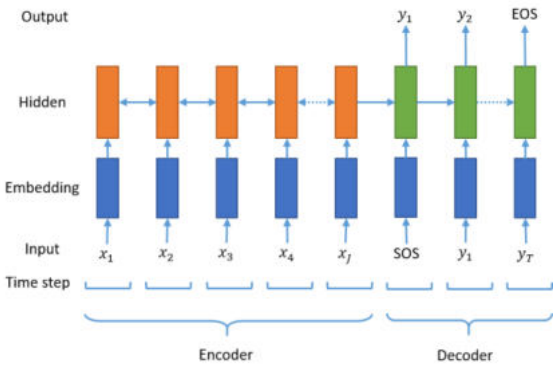

Figure 1. Seq2Seq model. (T. Shi, 2018:p.5)

of the decoder is the actual output not the previous predicted output. However, in model inference, the input will be the previous predicted output.

Standard RNN suffers from short term memory because of the vanishing gradient problem, where the gradient becomes very small during back-propagation that the RNN unit that gets this gradient to update won't learn. As a result, RNNs are difficult to train (Bengio et al, 1994). To solve this issue, LSTM (Long Short Term Memory) (Hochreiter and Schmidhuber, 1997) and GRU (Gated Recurrent Unit) (Cho et al., 2014) are used in sequence-to-sequence problems. LSTM and GRU are types of RNNs that have added gates units to control the flow and learn which information is important and which information to forget. LSTMs are similar to GRUs except that LSTMs use additional gate units and a separate memory cells that control the exposure of its memory content (state), whereas a GRU exposes the whole state. Chung et al. (2014) have compared the performance of LSTM and GRU, and they couldn't give concrete answer to which one performs better. Bahdanau et al. (2014) showed that both GRU and LSTM gives comparable results in machine translation task.

However, the fixed-size internal context vector limits the performance for the encoder-decoder networks, especially for long input sequences. To overcome this limitation, attention is used within sequences (Bahdanau et al. 2014, Luong et al. 2015). By using attention (see Figure 2) instead of having one context vector, a context vector is calculated at each decoding step as a weighted average of previous hidden states of the encoder. Later, this vector is fed into a feed-forward neural network layer along with the decoder hidden state to produce the predicted output for that step. Attention helps the decoder to know which parts of the input are important at each step.

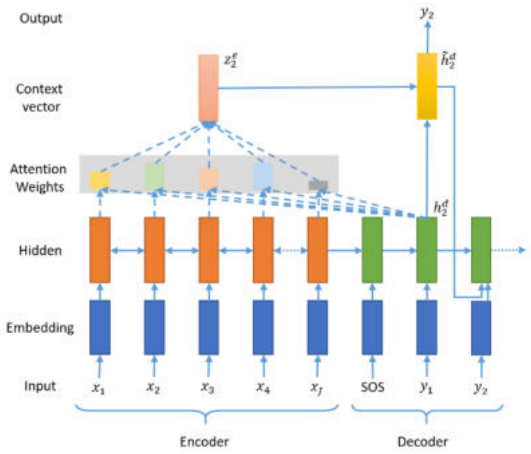

Figure 2. Seq2Seq with Attention. (T. Shi, 2018:p.6)

Previous studies show that the generating summaries suffer from several problems such as repetition and Out-Of-Vocabulary words (OOV), which are common in sequence-to-sequence natural language processing (NLP) problems. Different mechanisms have been added to enhance the encoder-decoder with the attention model. See et al. (2017) introduced a pointer-generator network to handle OOV words where the network uses a pointing mechanism to copy words from source documents into the output or generate novel words by a generator. Moreover, they applied the coverage mechanism into text summarisation problem where they used a coverage vector as input to the attention layer. The vector maintains a history of previous attention and thus prevents the model from repeating attention. To address the mentioned issues, we will adopt both pointer-generator and coverage mechanisms in our models.

C. Transformer Model

The Transformer model introduced by Vaswani et al. (2017), shown in Figure 3, consists mainly of an encoder and decoder. The encoder and decoder in turn consist of a stack of encoders and decoders over each other. The output of each encoder unit will pass as input to the next encoder and only the last encoder output will be fed into each decoder of the stack of decoders. Each encoder contains two parts, a self-attention layer and a feed-forward neural

network layer. While the decoder unit contains both units, it also has another attention layer in between.

As with most NLP tasks, the first step is embedding each input word into vectors. This embedding will be applied only to the input of the first encoder. Other encoders in the stack will take the output of the previous encoder as input. The embedded vectors will then be combined with positional encoding to encode the order of the input sequence. The positional embedding vectors for the input will enter the encoder, the first layer is a multi-head self-attention layer. Self-attention helps to understand the relations of input words and associate the current word to other related input words. The output of the self-attention layer is then passed to the feed-forward neural network, whose purpose is to strengthen the network and analyse more relation in the output of the multi-dead self-attention layer.

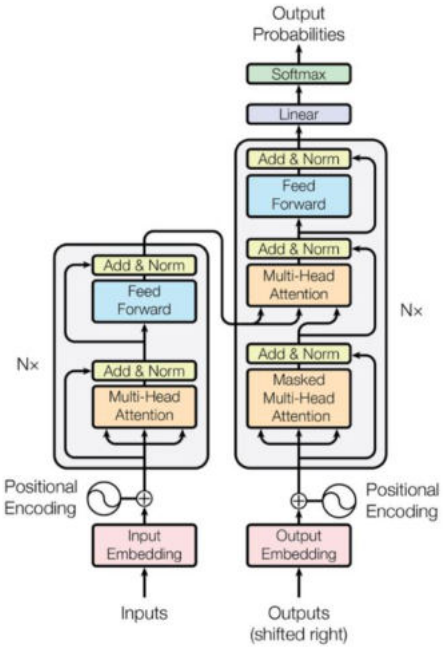

Figure 3. Transformer model (Vaswani et al., 2017:p.3)

The decoder unit consists of the two units of the decoder with an additional encoder-decoder attention layer. The first sub-layer is a masked multi-head self-attention to prevent the decoder from looking into future words in the output. The next sub-layer is the encoder-decoder multi-head attention which allows the decoder at each position to be aware of the words in the inputs; this layer provides the functionality of the encoder-decoder attention layer in the RNN sequence-to-sequence model. The final output after all the decoder units will be fed into a softmax to predict the next word with the highest probability.

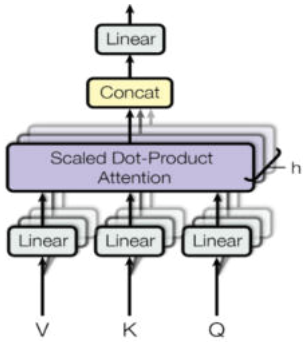

Figure 4. Multi-Head Attention (Vaswani et al., 2017:p.4)

Figure 4 shows the multi-head self-attention, where linear layers are used to calculate keys (K), queries (Q), and values (V) by learning three separated feed-forward neural network layers. The attention is calculated using the following equation:

$$\text{Attention}(\text{Q}, \text{K}, \text{V}) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V \quad (1)$$

Q is the matrix that represents a word in the input sequence; K is a vector representing all words in the input; and V is the vector representation for all the inputs. For the multi-head self-attention units in both encoder and decoder, V is similar to Q. However, they are different in the encoder-decoder attention layer in the decoder.

D. Discrete Metrics

Text generation models are trained using cross-entropy loss. The model produces probability distribution for the next tokens, the decoder generates a sequence based on this probability, and a summed loss from the whole sequence is used to redirect the model to learn to produce distributions closer to the actual distributions. However, NLP tasks are evaluated using discrete metrics like ROUGE (C.-Y. LIN, 2004), and BLEU (Papineni et al., 2002) which are non-differentiable and are used to judge the model after implementation.

C.-Y. LIN (2004) introduced ROUGE for summaries evaluation. ROUGE is a package of variants measurements such as ROUGE-L and ROUGE-N, where n stands for the number of n-grams. ROUGE variants measure the overlapping unigram (ROUGE-1), bigram (ROUGE-2), or longest matching subsequence (ROUGE-L) between the output and target summaries,

ROUGE-L doesn't only measure consecutive matching, but in-sequence matching. We will use ROUGE-1, ROUGE-2, and ROUGE-L variants in summaries evaluation.

To get a meaningful quantitative value, ROUGE measures the overlapping using precision P, and recall R. Recall in ROUGE indicates how much of the target summary the output summary covers, while precision measures the extent to which the output summary contains overlapped information. ROUGE also report F-score or F-Measure which is the harmonic mean of the precision and recall values. ROUGE scores are between 0 and 1. A ROUGE score of 0 means there is no overlap between output and target summaries.

$$\mathrm{P} = \frac{Number\ of\ overlapping\ words}{Total\ words\ in\ output\ summary} \quad (2), \quad \mathrm{R} = \frac{Number\ of\ overlapping\ words}{Total\ words\ in\ reference\ summary} \quad (3)$$

$$\mathrm{F} = 2 * \frac{P*R}{P+R} \quad (4)$$

E. Generalisation

Torralba and Efros (2011) suggested using cross-dataset generalisation to measure the bias of datasets. They measured how well a model trained on a dataset generalised when evaluated on other datasets. However, those datasets are from the same domain. They also introduced a cross-dataset performance drop *PD* as:

$$PD = \frac{Self - Mean\ others}{self} \quad (5)$$

Here, "Self" represents in-dataset performance where testing and training data are drawn from the same dataset, whereas "Mean others" is the average performance of datasets other than self.

Zhong et al. (2019) studied the effect of different dataset factors into generalisation behaviour of extractive summarisation models. The study addressed the dataset characteristics based on constituent and style factors. Constituent factors are positional and content values, while style factors are compression and density. They studied the influence of these factors by breaking down the test set into multiple sets, and by doing cross-dataset generalisation. The study shows that understanding dataset factors help in designing more reasonable models. In our study we will adopt their approach of breaking the test set based on different characteristics. Moreover, we will conduct cross-dataset generalisation.

Hendrycks et al. (2020) explored the generalisation ability by doing cross-dataset generalisation of different models' choices on different NLP tasks, such as bag-of-words models, ConvNets, LSTMs and Pretrained Transformers. The study shows that pretrained Transformers are more robust with Out-of-Distribution datasets. However, abstractive summarisation was not included in the NLP tasks studied.

F. Summary

Although several prior studies have focused on improving architectures for abstractive summarisation task, the generalisation ability of these architectures in abstractive summarisation has not been clearly explored. In this project, we will analyse the generalisation ability by considering the model choice and dataset's characteristics.

III. SOLUTION

The solution designed to implement the abstractive summarisation models is described in this section. The generalisation evaluation is also presented. To start, an overview of the solution and generalisation evaluation is given. Then, the implementation tools, datasets, detailed solution and generalisation evaluation are given in subsections.

A. *Solution Overview*

The Solution is divided into two parts: models' implementation and generalisation evaluation (see Figure 5). The models' implementation starts with preparing the training dataset, constructing the models, starting models training and hyper-parameters tuning until we obtain the final two abstractive models. The generalisation evaluation starts with preparing the cross-datasets and dividing the datasets based on different factors. Later, the implemented models will be used in the inference stage. The output summaries of inference will be evaluated based on different methods. Moreover, the topic evaluation will be done for both models with in-dataset and cross-dataset after clustering the datasets into different news categories.

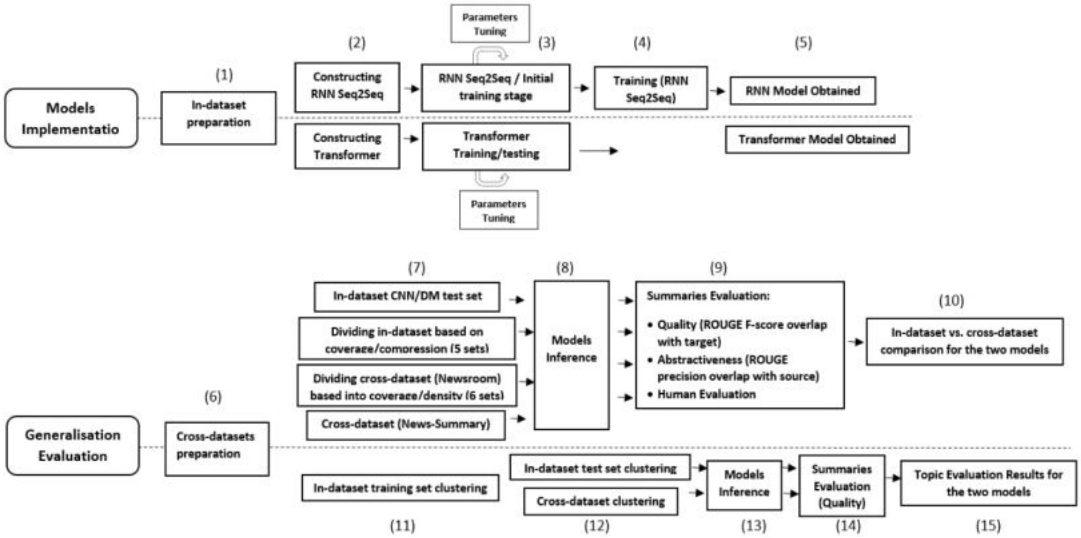

Figure 5. Solution Overview

B. *Implementation Tools*

For summarisation models implementation, an open-source toolkit OpenNMT[1] is used. OpenNMT is an open-source project for neural sequence learning released by Harvard NLP group and has been used in research and industry (Klein et al., 2017). We worked with OpenNMT-py which is a PyTorch[2] port of OpenNMT. The project provides three modules: pre-process, train, and translate (inference). Using OpenNMT made implementing the models easier and saved time for training and hyper-parameters tuning instead of constructing the models' architecture from scratch. Python language is used for data pre-processing, analysis, and further required implementation. Models training, inference, and evaluation have been done using NVIDIA CUDA Centre[3] (NCC) GPU system provided by the Department of Computer Science, Durham University. NCC provides a mix of GPUs: TITAN Xp, GeForce RTX 2080 Ti, and TITAN RTX. We used a single GPU which increased the training speed significantly.

C. *Datasets*

1) *Training dataset*

For training our models, we selected CNN/DM[4] dataset which has been used in most prior research for training/evaluating summarisation models (See et al, 2017). We chose this dataset

[1] OpenNMT Neural Machine Translation. https://opennmt.net/
[2] https://pytorch.org/
[3] http://community.dur.ac.uk/ncc.admin/
[4] CNN/DM. https://github.com/harvardnlp/sent-summary

for training the models since it is essential to compare and analyse models that are achieving the current state of the art results. The dataset consists of text-summaries pairs. We used the same split employed in prior research as follows: 287,226 training pairs, 13,368 validation pairs, and 11,490 testing pairs. We used the same pre-processing done by See et al. (2017). We truncated documents over 400 words since it has shown better performance than using the average text word length which is 800. Moreover, we truncated summaries of more than 100 words. Figure 6 shows the coverage and compression (Grusky et al., 2018) distributions of the training dataset, where **coverage** is the percentage of the words in the summary that's borrowed from the source input document, and **compression** is the ratio between the length of documents and summaries. The average coverage of the training dataset is 77.5% and the average compression is 13.8.

$$\text{coverage} = \frac{Number\ of\ intersection\ words}{length\ of\ summary} \quad (6), \quad \text{compression} = \frac{Length\ of\ Text}{length\ of\ Summary} \quad (7)$$

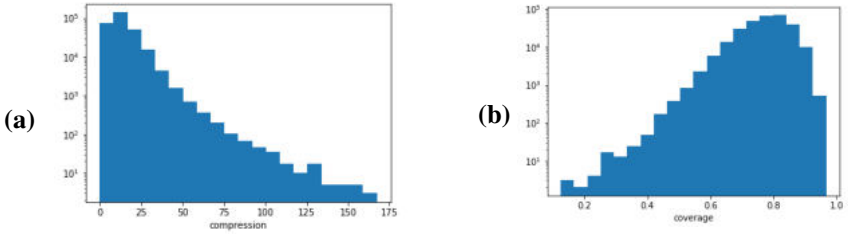

Figure 6. CNN/DM dataset distributions based on (a) Compression, (b) Coverage

2) Cross-datasets

We selected two datasets for cross-dataset evaluation. Both of our cross-datasets are from the news domain which is the same domain as our training datasets CNN/DM.

Newsroom[5] (Grusky et al., 2018) is a large dataset containing 1.3 million news articles and summaries. We chose this dataset because of its diversity of summarisation strategies (extractive and abstractive). The coverage and compression ratios of the dataset are 82.4 and 42.13 respectively. The Newsroom dataset provides statistical analysis for each text/summary pair, such as coverage, compression and density (Grusky et al., 2018), where density measures the percentage of the summary that could be described as extractive.

News-Summary[6] (Vonteru, 2017) is the second dataset. The dataset is small, containing 4395 text/summary pairs. It was mainly chosen since it was not evaluated in prior research. The coverage average of the summaries is 71% and the average compression ratio is 5.8.

D. Models Implementation

1) RNN Sequence-To-Sequence Model

Our training dataset is relatively large and it needs days to train the RNN sequence-to-sequence model. Therefore, we started implementation with an initial stage of limited training steps to tune the hyper-parameters. During parameters configuration, when one parameters is altered, other parameters remains unchanged in order to validate the accuracy of the tuned parameter.

There are different choices to select for RNN model architecture, such as the type of RNN, type of attention, number of layers, and number of hidden units. We started by studying RNN types, Vanilla RNNs, LSTMs, and GRUs. We eliminated vanilla RNNs since they are difficult to train. Both LSTM and GRU showed comparable results in literature, thus, we chose LSTM as they have additional memory cells. The next decision is whether to have RNN or bidirectional RNN (BRNN) (Schuster and Paliwal, 1997). BRNN is a modified type of RNN

[5] http://lil.nlp.cornell.edu/newsroom/index.html
[6] https://www.kaggle.com/sunnysai12345/news-summary?select=news_summary.csv

architecture that considers not just the past but also the future information. Thus, two propagations will be fed into RNN, forward and backward. Bidirectional RNNs play an important role to solve NLP problems where, in many cases, it's impossible to understand the correct meaning of the word without knowing the future words. As a result, we choose BRNN.

The next decision is choosing attention type. There are two types of attention to choose from: Bahdanau's attention (Bahdanau et al., 2014) and Luong's attention (Luong et al., 2015). To select between them, we trained the model on both for 100000 training steps and observed the accuracy from both. Figure 7 shows the accuracy for training and validation sets. The accuracy results of both types are similar, with Luong's attention resulting in slightly higher accuracy. ROUGE scores, (see Table 1), were calculated for both attention models output. We can see that ROUGE scores of both attention type are similar. However, Luong's attention requires less time to reach 100000 steps and since it doesn't use all last hidden states from the encoder, it requires less memory. As a result, we chose Luong's attention over Bandana's.

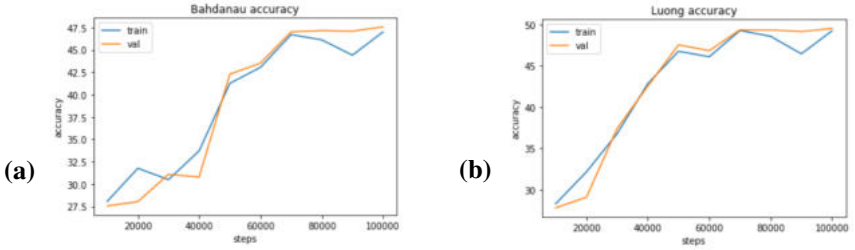

Figure 7. Model accuracy with (a) Bahdanau's Attention, (b) Luong Attention

The next step is tuning the number of hidden units. For initial training, a value of 256, 512, and 1024 were selected for testing. Figure 8 shows the results of accuracy until 100000 training steps. We can see that 512 and 1024 have higher accuracy than 256. However, both 512 and 1024 have very similar accuracy output. Since training with the 1024 units needs more time than 512 units, we only considered 256 and 512 models for summaries inference and calculating ROUGE values (see Table 2). As a result of ROUGE scores, we chose 256 for the number of hidden units since it takes less time to train with very similar ROUGE values.

Table 1. Bahdanau's vs Luong ROUGE

Attention	R-1	R-2	R-L
Bahdanau	0.281	0.102	0.285
Luong	**0.291**	**0.109**	**0.296**

Table 2. RNN Hidden units ROUGE

Hidden Units	R-1	R-2	R-L
256	0.291	0.109	0.296
512	**0.306**	**0.119**	**0.310**

Next, we studied the effect of the number of layers in our model. We trained our model until 100000 training steps with 1, 2, and 4 layers as shown in Figure 9. We can notice how the accuracy dropped with 4-layer model. Moreover, a model with one layer has higher accuracy than two layers and requires less training time. As a result, we chose one layer.

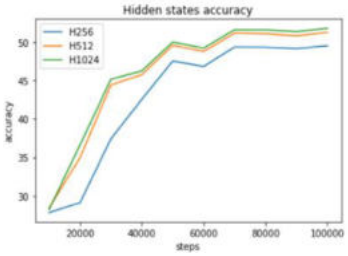

Figure 8. RNN Hidden Units scores

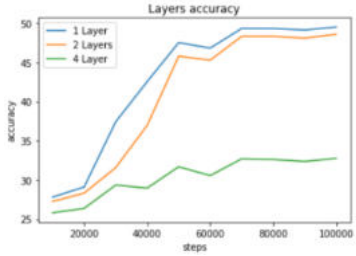

Figure 9. RNN Layers scores

We followed the work done by See et al. (2017) and Gehrmann et al. (2019). Thus, we set embedding size as 128. We didn't use pre-trained word embedding and let the model learn them. We used Adagrad (Duchi et al., 2011) as our Optimizer with a learning rate of 0.15. We performed gradient clipping at 2 for the gradient norm. We set the vocabulary size as 50000. We do not need to use a larger vocabulary size since we planned to add the pointer-generator

mechanism to our architecture. OpenNMT toolkit allows us to train additional copy attention layer which will support copying words from the input source documents. Adding this pointer-generator mechanism is important, since, without it, our vocabulary size should be more than 50000. Without copy mechanism, we would need a vocabulary larger than 150000 which will require training time to exceed four and five days to see proper results.

Next, we experimented with Dropout (Srivastava et al., 2014). Dropout is a regularisation method used in Deep Learning to prevent over-fitting by randomly dropping off hidden units. For dropout, a value is selected between 0 and 1 to act as a probability for the number of units to be disabled after updating weight. We trained the model with four different values of dropout (0.0, 0.1, 0.3 and, 0.5) and observed the accuracy as shown in Figure 10. We can see that the accuracy for 0.0, 0.1, and 0.3 are very similar, while there is a small drop in the accuracy with dropout equals to 0.5. Then we selected the model with 0.3 to further evaluate by generating output summaries and calculating ROUGE scores (see Table 3). We noticed that ROUGE scores are even slightly higher when dropout equals 0.3.

Table 4 summarises our RNN sequence-to-sequence model choices and tuning results. Our final model contains 42525525 training parameters. We then trained the model until it reached 300000 training steps for 26 hours. The model converged around 50% accuracy achieving a validation accuracy of 51%.

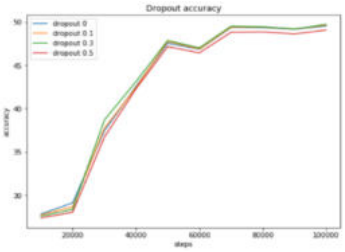

Figure 10. Dropout accuracy

Table 3. Dropout ROUGE

Dropout	R-1	R-2	R-L
0.0	0.291	0.109	0.296
0.3	**0.297**	**0.111**	**0.3**

Table 4. RNN Seq2Seq parameters

Parameter	Value
RNN	Bidirectional LSTM
Embedding size	128
Batch size	16
Attention	Luong
Hidden size	256
Layers	1
Dropout	0.3
Optimizer	Adagrad
Learning rate	0.15
Clipping	2

2) Transformer Model

Transformer implementation by OpenNMT is done by setting the encoder and decoder type to "Transformer" and adjusting the parameters. First, we studied and tuned the architectural parameters for the Transformer, which are the number of layers, number of heads, embedding size, and the type of self-attentions. Starting by tuning the number of layers, we trained the model for 100000 training steps with 4, 6, and 8 layers (see Figure 11). We noticed that a model with 4 or 8 layers performed better than 6 layers, although the original proposed Transformer was constructed with 6 layers. We chose a 4-layer model since it requires less training time. Then we tuned the feed-forward hidden units, compared the accuracy of 512 and 256 hidden units (see Figure 11), and as a result, we selected 512 units. We chose the number of heads to be 8 and set self-attention type to "scaled dot product" as proposed by the original Transformer.

Moreover, we followed the original paper (Vaswani et al., 2017) and Gehrmann et al. (2019) with the selection of training parameters, optimizer, learning rate, Label smoothing (Szegedy et al., 2016), and dropout. Moreover, we set the vocabulary size to 50000 and applied the pointer/generator architecture which will solve the out-of-vocabulary and rare data problem. Table 5 summaries our final choices for the Transformer parameters. Our model contains 80682325 training parameters and it was trained for 100000 training steps for about 2 days. The model converged around 60% accuracy achieving 55.9% validation accuracy. Moreover, the model passed 50% accuracy after 10000 training steps. We didn't train the model further more than 100000 steps since the ROUGE scores were comparable with our RNN sequence-to-sequence model. Thus, we moved into further analysing the Transformer model after 2 days of training.

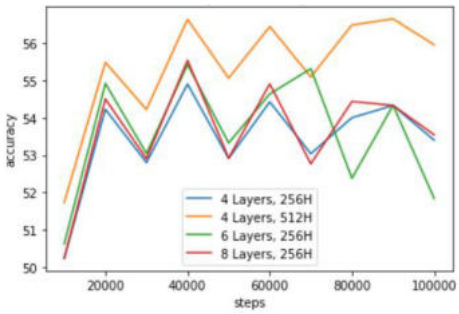

Figure 11. Transformer Layers, hidden units accuracy

Table 5. Transformer parameters

Parameter	Value
Embedding size	512
Feed forward size	512
Batch size	4096
Attention	Scaled dot product
Layers	4
Heads	8
Dropout	0.1
Label smoothing	0.1
Optimizer	Adam
Learning rate	2

E. Models Inference

Inference is the process of prediction using the trained model. Here we used our training models RNN sequence-to-sequence and Transformer to generate summaries from the test datasets.

During Inference, the decoder will generate the most likely word at each step based on the likelihood in what is known as the greedy search (Goldberg, 2017:227). Although the greedy search is fast, it is not optimal and might not generate the best sequence. Another option is using the Beam Search algorithm (Freitag and Al-Onaizan, 2017). The Beam Search selects the most likely N-words at each step, where N is the Beam width, and keeps all the possible sequences in memory. The higher the value of the Beam width, the better performance we achieve in generating the output sequence. However, it also means slowing the decoding process. We selected a value of 5 for the Beam width.

Additionally, we adopted several mechanisms to handle common issues in text generation models during inference. First, we applied the coverage mechanism (See et al., 2017) to eliminate repetition in the generated sequence by preventing attention on the previously generated words. We also applied the length penalty by Wu et al. (2016) to encourage generating longer sequences during beam search. Moreover, we set the minimum length of the generated summaries depending on the test dataset, and the maximum summary length as 100.

F. Generalisation Evaluation

After completing the models' implementation, we moved to evaluating the models to answer our question *"To what extent do abstractive summarisation models have Generalisation ability?"*. We evaluated the models with several evaluation methods in different steps. First, we did in-dataset evaluation. Second, we proceeded to the cross-dataset evaluation. Finally, we studied the models' ability to handle topic bias and imbalanced categories.

1) In-dataset Evaluation

At this stage, we evaluated the model with the test set of CNN/DM. Then we broke up the test set based on Coverage and Compression to analyse the behaviour of our models with different dataset factors. We calculated the coverage and compression for each document/summary pair and using the same thresholds in the Newsroom dataset. We divided the pairs into low, medium and high sets based on coverage and compression. As for the coverage thresholds, a summary with coverage less than 0.8 is considered to have low coverage, and less than 0.95 is considered to have medium coverage. As a result our test sets were divided into low and medium coverage sets containing 6382 and 5108 pairs respectively. Moreover, a summary with more than 35 compression ratio is considered to have high compression and with more than 15 ratio is considered to have medium compression. Thus, the test sets were divided into low, medium, and high sets containing 8254, 3044, and 192 pairs respectively.

2) Cross-dataset Evaluation

After in-dataset evaluation, we evaluated the models' ability to handle cross-datasets. Cross-dataset is a dataset that is not part of model training and has different distribution than the training dataset. However, the cross-dataset is in the same domain as the training dataset. All of our datasets are from the News domain. We chose two datasets for the cross-datasets evaluation, Newsroom and News-Summary. News-Summary is relatively a small dataset that is used for cross-dataset evaluation. However, Newsroom is a huge and diverse dataset containing pairs of documents/summaries with different characteristics based on density, coverage, and compression. We evaluated the models based on density and, for that, we divided the test dataset into abstractive, mixed, and extractive sets containing 35349, 35915, and 36165 pairs respectively, and conducted an evaluation on the resulting sets. Then, we split the test dataset based on coverage into low, medium, and high sets containing 34986, 36088, and 36355, and evaluated our models on these sets.

3) Topic Bias Evaluation

For this stage, we investigated the influence of topic bias on the ability of abstractive models. By topic bias we mean the imbalance of dataset topics in our training dataset. Since the domain of our datasets are News, we studied the behaviour of the models with different news categories, such as sport, crime, and economy.

The first step is analysing the categories in the training dataset. The dataset doesn't give us the information about the categories or explicit labelling for documents. Therefore, to classify these news documents into different categories we used clustering document algorithms. Clustering is an unsupervised learning method to assign a document into a cluster so that documents inside the cluster are similar and documents in different clusters are dissimilar. There are different algorithms for document clustering; we used K-means algorithm (Jain, 2010). K-means partitions the data into k clusters, where k is a user-defined parameter, the algorithm defines k centroids, and the idea is that each object in the cluster is similar and close to the cluster's centroid. We used *tf.idf* representation for our documents (Bafna et al., 2016), where *tf* is the term frequency which measures the count of the term in the document, whereas *idf* is the inverse document frequency of a given term. *Idf* is high with rare terms and low with common terms repeated in many documents. With *tf.idf* we assign a weight composed of *tf* and *idf* for each term in the document equals to *tf* x *idf*. Moreover, we used stemming to convert words to their base form and removed non-letter words and stop words which are the common words in a language that do not carry meaningful information (Thompson A., 2017).

With CNN/DM, we used K-means to classify the documents in the dataset into different categories. We used Elbow method to find the optimal number of clusters K. Elbow method, calculates the Within-Cluster-Sum of squared (WSS) for a range of values of K and chooses K for which the WSS starts to diminish. For our experiments with K, we chose MiniBatchKMeans (Sculley, 2010) to find the clusters for each K since it requires less time because it uses fixed random batches of data instead of using the whole dataset as in K-means. Then we applied K-means to find the clusters for the optimal value of K. In the plot of K vs. WSS (see Figure 12), WSS kept decreasing with increasing K; however, when trying to find clusters for K more than 14, the clusters were about specific topics and not general categories, we want to identify topics as sport and crime. Thus, we experimented K with 8 and 11. We found that a value of 11 is able to identify more general topics. For example, the topic "economy and technology" wasn't clear with K equals to 8. Thus, we chose K as 11 in CNN/DM dataset. Figure 13 shows some of the clusters considered in the training dataset. We noticed that the training data contains a higher percentage of the crime category than other

topics. Then, we found the clusters of the test set of CNN/DM. Finally, we used the Elbow method with the Newsroom mixed set (see Figure 14), selected K as 11, and applied K-means to find the clusters.

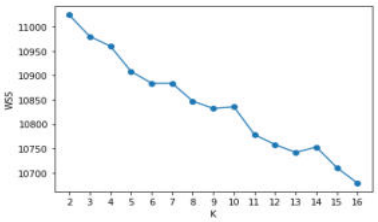

Figure 12. CNN/DM WSS vs. K

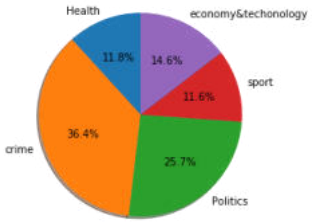

Figure 13. Training Data Clustering

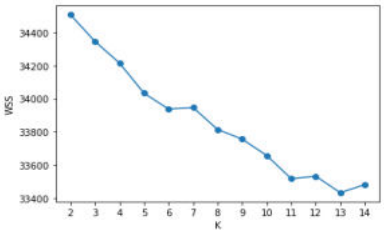

Figure 14. Newsroom WSS vs. K

IV. Results and Evaluation

A. *Summaries Evaluation Methods*

Evaluating abstractive summaries is challenging since there are several aspects that must be evaluated of the output summary, such as its quality, readability, and abstractiveness. We adopted different methods to evaluate summaries generated from our abstractive models.

1. Quality: We evaluate the quality of the output summaries in terms of ROUGE metric. We measured the overlap between the output summaries and target (reference) summaries. Moreover, we used three variants of ROUGE F-scores, ROUGE-1, ROUGE-2, and ROUGE-L. Higher values of ROUGE-1 mean that our summaries are covering more words of the target summaries. ROUGE-L indicates that our summaries overlapped words are in-sequence and have similar order with words in target summary. Having higher ROUGE-1, ROUGE-2, and ROUGE-L values indicates that our summaries are fluent.
2. Abstractiveness: Measuring abstractiveness is essential when evaluating abstractive models. Abstraction is the ability to generate novel n-grams terms. This means measuring if the output summaries contain different combinations of the words in source documents. When measuring novelty, we do not only measure new words in output summaries but also words combinations. If words are copied without modifications then the model will be doing an extractive work rather than abstractive and novelty will be very low or zero. To measure abstractiveness, we calculated the ROUGE precision overlapping scores between output summaries and source using ROUGE-1, ROUGE-2, and ROUGE-L variants.
3. Human Evaluation: Several examples were studied and analysed from both models output summaries manually. Readability and relevance were the most qualitative metric evaluated.

B. *Generalisation Evaluation*

1) *In-dataset Results*

When evaluating CNN/DM test dataset output from both RNN and Transformer models with reference summaries (see Table 6), we can see that both models have similar ROUGE scores and the Transformer ROUGE scores are slightly higher. ROUGE-1 scores imply that roughly 35% of summaries' words are overlapped with reference summaries. Additionally, ROUGE-2 scores mean that the model's output summaries have around 15% bigram overlap with the reference summaries. Finally, ROUGE-L measures the longest matching sequence between output and target summaries, around 35% of the overlapped words have the same order as in

target summaries. The results we get from ROUGE scores implies that both models are able to produce readable, fluent, and informative summaries.

Figure 15 shows the ROUGE precision scores when evaluating the reference summaries and the models' output summaries with source documents, one can notice from the models' output ROUGE-1 scores that more than 90% of summaries' words with both models are extracted from source. Moreover, the high value of ROUGE-L implies that the content of the summaries appears in the same order as they appeared in the source. Thus, there is a low novelty in generating words and ordering. An interesting value here is ROUGE-2 score; we can see the difference in ROUGE-2 value between our models' output summaries and the reference summaries. The reference summaries have less overlapping with source input and thus, are more abstractive. High values for ROUGE-2 implies that summaries generated have small bigrams novelty. However, high values of ROUGE-2 imply that output sentences are readable since they represent the bigrams overlap between summaries and source. In Addition, we notice that ROUGE-2 value for RNN model is higher than it is in the Transformer model. Thus, the Transformer's summaries are more abstractive than RNN summaries and generates higher novel bigrams.

Table 6. CNN/DM ROUGE f-scores with target summaries

Model	R-1	R-2	R-L
RNN	0.341	0.146	0.341
Transformer	**0.342**	**0.148**	**0.346**

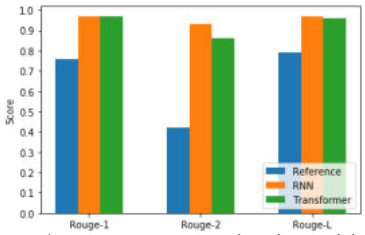

Figure 15. CNN/DM ROUGE evaluation with source

Table 7 below shows the ROUGE scores overlap with reference summaries after breaking the test set based on coverage and compression. The difference in ROUGE values between low and medium coverage sets is clear in both models. The drop with low coverage which is reasonable since we noticed from Figure 6 that our training dataset is skewed towards medium coverage. Moreover, when we analysed compression results, we notice the significant drop between low and high compression sets. Generating highly compressed summaries is challenging since summaries must be very concise. Our training dataset is skewed towards low compression which explains the significant drop. However, ROUGE scores in high compression set imply that summaries are still informative and have around 25% overlap words with target summaries. Table 8 shows an example of a generated summary form both models.

Model	Metric	Low	Medium
RNN	R-1	0.309	**0.383**
	R-2	0.11	**0.19**
	R-L	0.307	0.386
Transformer	R-1	0.311	0.381
	R-2	0.116	0.188
	R-L	0.312	**0.388**

Table 7 (a). ROUGE f-score coverage evaluation

Model	Metric	Low	Medium	High
RNN	R-1	**0.357**	0.307	0.239
	R-2	0.159	0.118	0.083
	R-L	0.358	0.305	0.236
Transformer	R-1	**0.357**	0.307	0.241
	R-2	**0.16**	0.119	0.084
	R-L	**0.362**	0.308	0.237

Table 7 (b). ROUGE f-score compression evaluation

Table 8. CNN/DM summary example

Reference robert lewis burns jr. was part of lynyrd skynyrd 's original lineup . his car hit a mailbox and a tree just before midnight .
RNN burns , 64 , died after his car hit a mailbox and a tree in cartersville . no other cars were involved in the crash , which occurred shortly before midnight . burns was part of the genre-defining band 's original lineup .
Transformer robert lewis burns , 64 , died after his car hit a mailbox and a tree in cartersville . no other cars were involved in the crash , which occurred shortly before midnight .

2) *Cross-dataset Results*

a) *Newsroom Dataset*

Table 9 shows the ROUGE scores after evaluating the Newsroom dataset divided based on density with their corresponding target summaries. Scores for abstractive summaries are small for both models. Whereas scores increased for mixed and extractive sets. That increase means that both models are generating the same words found in the source documents. However, the difference between ROUGE f-scores for RNN model and Transformer model with extractive summaries set is big. This means that RNN model is generating less abstractive summaries than the Transformer. Finally, RNN model scores for extractive summaries set are even higher than in-dataset evaluation, especially the ROUGE-2 score, whereas the Transformer model scores are smaller than in-dataset scores. This high increase in scores with the extractive summarisation set indicates that RNN model is doing an extractive work.

Table 9. Newsroom – ROUGE f-scores evaluation with reference

Dataset	Model	R-1	R-2	R-L
Newsroom Abstractive	RNN	0.114	0.013	0.104
	Transformer	0.109	0.01	0.102
Newsroom Mixed	RNN	0.207	0.082	0.196
	Transformer	0.182	0.053	0.173
Newsroom Extractive	RNN	**0.399**	**0.309**	**0.41**
	Transformer	0.27	0.137	0.265

Figure 16 shows the scores of evaluating output summaries of the mixed test set with source documents. High ROUGE precision scores show that summaries are generated using words from input with less novel words. RNN model scores with source inputs have higher overlapping than Transformer's summaries which indicates that RNN model is copying terms from source input with less novelty than the Transformer. Table 10 shows ROUGE scores when dividing the test dataset based on coverage. ROUGE scores are higher with high coverage set.

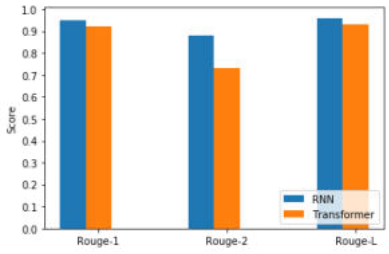

Figure 16. Newsroom ROUGE evaluation with source

Table 10. Coverage evaluation with reference summaries

Model	Metric	Low	Medium	High
RNN	R-1	0.121	0.224	**0.349**
	R-2	0.022	0.108	**0.253**
	R-L	0.112	0.218	**0.358**
Transformer	R-1	0.116	0.191	0.257
	R-2	0.015	0.06	0.122
	R-L	0.109	0.182	0.252

An example of a summary generated is shown in Table 11. We can notice that the Transformer summary is more abstractive and concise, whereas RNN model is doing more extractive than abstractive. Finally, The Transformer output summaries of Newsroom have more UNKNOWN words than RNN. For example, around 0.78% of words generated from Newsroom mixed set are UNKNOWN with the Transformer model while RNN generates 0.03% UNKNOWN.

Table 11. Newsroom mixed example

Reference Alexa has added more than 900 skills in the past six months.
RNN virtual assistant, Alexa, has gotten a whole smarter in the past six months. tech giant boasted on Friday that Alexa now has 1,000 skills. That's up from just over one hundred skills .
Transformer Alexa has 1,000 skills. That's up from just over one hundred skills . developers are learning about the service, bringing useful and innovative skills .

b) News-Summary Dataset

ROUGE scores after evaluating the output of the models on News-Summary dataset with target summaries are shown in Table 12. We see that ROUGE F-scores for RNN model are higher than the Transformer model. This means that RNN output summaries are more overlapped with target summaries. In Figure 17, we evaluated the output summaries with source input. We notice from ROUGE-1 and ROUGE-L precision-scores that more than 90% of words in output summaries from both models existed in source input and in the same order. ROUGE-2 precision values are higher in RNN model, with 90% bigrams overlap to 74% bigram overlap in the Transformer, which implies that the Transformer output summaries are more abstractive. The high values of ROUGE precision scores in Figure 17 indicate that summaries are fluent and readable. An example of a summary generated in Table 13 shows that RNN model is copying the words as they are found in the source input, and stopped when it reached the maximum length.

Table 12. News-Summary output evaluated with target

Model	R-1	R-2	R-L
RNN	**0.35**	**0.15**	**0.32**
Transformer	0.27	0.09	0.26

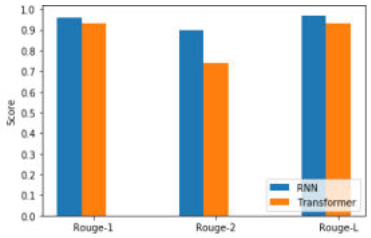

Figure 17. News-Summary ROUGE evaluation with source

Table 13. News-summary example

Reference A 60-year old Dalit woman was allegedly lynched in Agra after villagers thought that she was behind the recent cases of chopping hair of sleeping women. The family members of the woman who left home around 4 am on Wednesday said, "She pleaded that she had lost her way but they took her as one responsible for chopping women hair."
RNN A 60-year old Dalit woman was allegedly lynched in Agra after villagers thought she was out to cut the hair of sleeping women, the first reported …….instances of sleeping women being attacked and their hair chopped off short. Family members said a case was registered including charges relating to atrocities against Dalits, and teams keeping a vigil in the area where the woman was killed. The police also appeared
Transformer mass hysteria.Growing number of sleeping women attacked and hair chopped off . short. daughter-in-law Kalpana Jatav.Local police denies death was connected. people spotted her. She pleaded that she had lost her way

3) Topic Bias Evaluation Results

a) CNN/DM

Figure 18 shows ROUGE-L f-scores for different topic categories from CNN/DM test set with their referenced summaries. Results for categories are similar in RNN and Transformer models. This indicates that both models are capable of handling imbalanced categories found in the training dataset. We can see that ROUGE f-scores are slightly higher in crime category than sport and politics. Those slight differences might be because of the bias in the topics in the dataset. However, the difference is not big enough to say that models are affected.

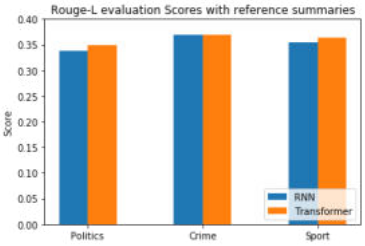

Figure 18. CNN/DM Topic evaluation

b) Newsroom

The different categories chosen for evaluation from the Newsroom mixed dataset are Economy, Technology, Crime, Sport, and Politics. Figure 19 shows the ROUGE-L f-scores for the

models' output summaries of these topic sets when evaluated with their corresponding reference summaries. We notice that ROUGE scores are in the same range among these categories. Moreover, we can see slightly higher scores in the crime topic set, but the difference is very small and doesn't indicate that the models' output is affected by the topic imbalanced.

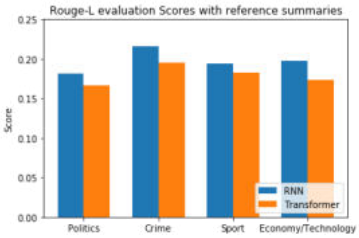

Figure 19. Newsroom Topic evaluation

C. In-dataset vs. Cross-dataset

Table 14 shows the performance drop, calculated using Eq. (5), of ROUGE-L evaluation score with reference summaries between in-dataset (self) and cross-datasets. We used the results of Newsroom mixed, Newsroom abstractive, and News-Summary datasets to calculate the mean performance of other cross-datasets. We can notice the significant drop in ROUGE scores for both models which demonstrates that out-of-distribution is quite serious in abstractive summarisation, thus challenging the current state of the art models where the training and testing data are usually drawn from the same distribution. The performance of the Transformer is slightly higher for in-dataset evaluation, whereas, its performance drop in ROUGE scores is higher than RNN Seq2Seq model. However, abstractiveness and manual evaluation demonstrated that RNN model is producing more extractive summaries and this explains the better performance with ROUGE scores for RNN model.

Figure 20 shows the abstractiveness evaluation for in-dataset and cross-datasets. We noticed that both models have low novelty in general. We can see that CNN/DM ROUGE-2 overlap is higher than cross-datasets for both RNN and Transformer which is expected since it has the same distribution as the training data. The Transformer abstrativeness increased with cross-datasets and had less ROUGE-2 overlapping scores, while the RNN model ROUGE-2 scores slightly decreased. Our manual analysis demonstrated that RNN has extractive behaviour and is relying more on the pointer/generator mechanism with out-of-distribution data. This explains why the RNN model's output summaries have less percent of UNKNOWN words with cross-datasets in comparison with the Transformer's output.

Table 14. Summary Quality comparison

Model	self	Mean others	Drop%
RNN	0.341	0.20	41.3%
Transformer	0.346	0.178	48.5%

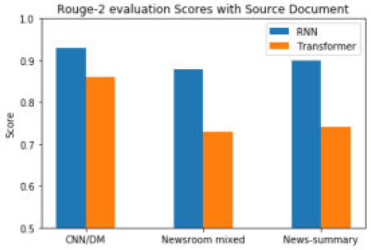

Figure 20. Abstractiveness comparison

D. Solution Strengths

Our research question is "*To what extent do abstractive summarisation models have generalisation ability?*". The solution addressed the generalisation behaviour by implementing two of the most well-known architectures – RNN sequence-to-sequence and Transformer. While implementing these models, we studied the behaviour of different configuration parameters and selected parameters that achieved better accuracy and evaluation scores. We also adopted several mechanisms in our models to eliminate known issues for summarisation such as coverage mechanism to handle repetition, length penalty to encourage the model to generate longer sentences, pointer/generator network to allow the model to copy words from source documents to solve unknown and out of vocabulary words, and Beam Search in models inference stage to ensure that our decoder is generating the best output sequence.

The next stage was analysing generalisation and the behaviour of our models, we selected different behaviour levels to evaluate, in-dataset, cross-dataset, and dataset topic bias. By doing this analysis we evaluated the ability of our models to handle cross-dataset and out-of-

distribution data. Moreover, the dataset topic bias evaluation demonstrated whether the models are affected by imbalanced topics categories in the training dataset. Additionally, the evaluation addressed different criteria such as summaries quality, abstractiveness, and manual human analysis, which provides a better insight into the output summaries generated. During in-dataset and cross-dataset, we split the datasets based on different characteristics, such as density, coverage, and compression. Then, we evaluated the models based on these factors. By doing that, we analysed the effect of our training dataset on the model's output.

E. Solution Limitations

When selecting parameters for models' implementation, hyper-parameters tuning was limited to several parameters and didn't include all configurable parameters. Some hyper-parameter tuning was done using a fixed range of possible values. Whereas, some parameters' values were selected by referring to the previous implementation in literature or by relying on the library default values. Our models' results might be affected if these values were tuned.

Moreover, models training time is an important factor in the models' ability. We trained the RNN sequence-to-sequence model for 300000 training steps and the Transformer's training was till 100000 training steps. Further increasing training time might improve the results of ROUGE scores. Additionally, during abstractiveness evaluation of the output summaries, we used three variants of ROUGE where it is also possible to use other variants to strengthen our results. Furthermore, our manual analysis was limited; it would have given better insight if we had asked different participants to do more human evaluation as a part of a survey.

F. Threats to Validity

One threat to **construct** validity might be the suitability of the evaluation parameters. To address this threat, we used three variants of the ROUGE metric for evaluating the summary quality and abstractiveness. When working with summary quality, we used ROUGE-1, ROUGE-2, and ROUGE-L, and we reported F-measure which captures both precision and recall. These metrics have been exhaustively used for evaluating the quality of the summary in literature. Moreover, when evaluating abstractiveness, we used ROUGE-1, ROUGE-2, and ROUGE-L precision values to measure the overlap between the summary and text.

One threat to **internal** validity is the hyper-parameters tuning in the selected models. To address this threat, since we couldn't do parameters tuning with all possible values, some of these parameters used values from literature where it gave good results. For other parameters, we tuned with a fixed range of values.

External validity deals with the generalisation of our results. To address this threat, first, we selected two famous different architectures for summarisation models. Most rapid improvements in literature are built on these models such as using reinforcement learning to further train our model. Second, we used two datasets for cross-dataset testing. The two datasets have different characteristics, and one of them - the Newsroom - is a very diverse dataset.

V. CONCLUSION

In this project, we studied the generalisation ability for Deep Learning abstractive summarisation models. In order to do that, we implemented two current states of the art architectures: RNN sequence-to-sequence and Transformer models. Different mechanisms and design ideas were adopted into our architectures to address known issues in summarisation models such as repetition and out of vocabulary problems. After implementation, we moved into studying models' generalisation. We analysed models' output for in-dataset and different

cross-datasets. Moreover, we studied the output of models based on different factors in datasets such as compression, coverage, and density. Finally, we used unsupervised learning to split our datasets into different topic categories and investigated how abstractive models are affected by imbalanced categories in the training dataset.

We demonstrated that both models are able to handle in-dataset efficiently. However, with cross-datasets, the RNN sequence-to-sequence is doing more extractive than abstractive work, while the Transformer output summaries are more abstractive. Thus, the Transformer outputs are more concise while the RNN outputs contain unnecessary information as a result of its extractive behaviour. Moreover, the RNN model is sometimes not able to generate a clear summary and stop abruptly. Whilst this behaviour could be found in the Transformer, it was more common with the RNN model. Additionally, both RNN and the Transformer models are not clearly affected by imbalanced topics in the training dataset. Finally, both models were affected by the training dataset and produced outputs that have better ROUGE evaluation with higher coverage and lower compression datasets.

Our evaluation process highlighted the need for other evaluation methods other than ROUGE for abstractive summaries. For cross-dataset evaluation, ROUGE scores were better with RNN model output; however, our manual analysis and novelty evaluation shows that RNN model is doing less abstractive work and is sometimes unable to generate meaningful summaries. ROUGE overlapping scores between output and target summaries are heavily used in literature to evaluate summaries qualities. However, conducting abstractiveness and Human evaluation is essential to judge abstractive models' abilities, especially with cross-datasets.

Since the Transformer model has better generalization ability, future work may investigate the model's behaviour if evaluated with out-of-domain data. Moreover, it would be essential to study the generalisation ability of the model if trained with different domains other than the news domain. Besides, increasing vocabulary size and training time will remove the out-of-vocabulary problem found with the Transformer's output summaries in cross-datasets. Finally, our study only addressed neural networks architectures for the abstractive summarisation task. However, recently reinforcement learning has been combined with neural network architectures to improve the performance of Sequence-to-Sequence NLP tasks such as translation and summarisation (Paulus et al., 2017). It would be interesting to explore generalisation in abstractive summarisation after combining reinforcement learning with our architectures.

REFERENCES

Bafna P., Dhanya p., and Vaidya A., (2016). "Document clustering: TF-IDF approach", in *Proc. of International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pp. 61-66.

Bengio Y., Simard P., and Frasconi P., (1994). "Learning long-term dependencies with gradient descent is difficult", *IEEE Transactions on Neural Networks*, 5(2):157–166.

Bahdanau D., Cho K., and Bengio Y., (2014). "Neural machine translation by jointly learning to align and translate", in *Proc. of ICLR 2015*.

Bahdanau D., Chorowski J., Serdyuk D., Brakel P., and Bengio Y., (2016). "End-to-end attention based large vocabulary speech recognition", in *Proc. of 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4945–4949.

Cho K., van Merrienboer B., Gulcehre C., Bougares F., Schwenk H., and Bengio Y., (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation", in *Proc. of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, pp. 1724-1734.

Chung J., Gulcehre C., Cho K., and Bengio Y., (2014). "Empirical evaluation of gated recurrent neural networks on sequence modelling", in *NIPS 2014 Workshop on Deep Learning, December 2014*.

Duchi J., Hazan E., and Singer Y., (2011). "Adaptive subgradient methods for online learning and stochastic optimization", *Journal of Machine Learning Research*, 12:2121–2159.

Freitag M., Al-Onaizan Y., (2017). "Beam Search Strategies for Neural Machine Translation", in *Proc. of the First Workshop on Neural Machine Translation,* pp. 56–60

Gehrmann S., Deng Y., and Rush A., (2019). "Bottom-Up Abstractive Summarization", in *Proc. of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4098-4109.

Goldberg Y., (2017). *Neural Network Methods in Natural Language Processing*. Available at: https://ieeexplore.ieee.org/document/7909255 (Accessed: 31 August 2020).

Grusky M., Naaman M., and Artzi Y., (2018). "NEWSROOM: A DATASET OF 1.3 MILLION SUMMARIES WITH DIVERSE EXTRACTIVE STRATEGIES", *arXiv preprint arXiv:1804.11283*.

Gupta S. and Gupta S.K. (2018). "Abstractive Summarization: An Overview of the State of the Art. Expert Systems with Applications", *Expert Systems with Applications*, 121:49-65.

Hendrycks D., Liu X., Wallace E., Dziedzic A., Krishnan R., and Song D., (2020). "Pretrained Transformers Improve Out-of-Distribution Robustness", in *Proc. of the 58th Annual Meeting of the Association for Computational Linguistics*.

Hochreiter S. and Schmidhuber J., (1997). "Long Short-Term Memory", *Neural Computation*, 9(8):1735–1780.

Jain A. K. (2010) "Data clustering: 50 years beyond K-means", *Pattern Recognition Letters*, 31(8):651–666.

Klein G., Kim Y., Senellart J., and Rush A., (2017). "OpenNMT: Open-Source Toolkit for Neural Machine Translation", in *Proc. of ACL 2017, System Demonstrations*, pp. 67-72.

LIN C.-Y., (2004). "ROUGE: A package for automatic evaluation of summaries", in *Proc. of Workshop on Text Summarisation Branches Out*. pp. 74–81.

Luong T., Pham H., and Manning C. D., (2015). "Effective approaches to attention-based neural machine translation", in *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421.

Moratanch N. and Chitrakala S., (2017). "A survey on extractive text summarisation", in *Proc. of the 2017 International Conference on Computer, Communication and Signal Processing (ICCCSP)*, pp. 1-6.

Papineni K., Roukos S., Ward T., and Zhu W., (2002). "BLEU: a Method for Automatic Evaluation of Machine Translation", in *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318.

Paulus R., Xiong C., and Socher R., (2017). "A deep reinforced model for abstractive summarisation", *arXiv preprint arXiv: 1705.04304*.

See A., Liu P. J., and Manning C. D., (2017). "Get to the point: Summarisation with pointer generator networks", in *Proc. of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1073–1083.

Schuster M. and Paliwal K.K., (1997). "Bidirectional recurrent neural networks", *IEEE Transactions on Signal Processing*, 45(11):2673–2681

Sculley D., (2010). "Web-scale k-means clustering", in *Proc. of the 19th international conference on World wide web (WWW '10). Association for Computing Machinery*, pp. 1177–1178.

Shi T., Keneshloo Y., Ramakrishnan N., and Reddy C. K., (2018). "Neural Abstractive Text Summarisation with Sequence-to-Sequence Models: A Survey", *arXiv preprint arXiv: 1812.02303*.

Srivastava N., Hinton G., Krizhevsky A., Sutskever I., and Salakhutdinov R., (2014). "Dropout: A simple way to prevent neural networks from overfitting", *The Journal of Machine Learning Research*, 15(1):1929–1958.

Szegedy C., Vanhoucke V., Ioffe S., Shlens J., and Wojna Z., (2016). "Rethinking the Inception Architecture for Computer Vision", in *Proc. of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 2818-2826.

Thompson A., (2017). All the news. Available at: https://towardsdatascience.com/all-the-news-17fa34b52b9d. (Accessed: 31 August 2020).

Torralba A. and Efros A. A., (2011). "Unbiased look at dataset bias", *CVPR 2011*. pp. 1521-1528.

Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A., Kaiser L., and Polosukhin L., (2017). "Attention Is All You Need". *arXiv preprint arXiv:1706.03762*.

Vonteru K. (2017). NEWS SUMMARY, Version 2. Available at: https://www.kaggle.com/sunnysai12345/news-summary?select=news_summary.csv. (Accessed: 31 August 2020).

Wu y., Schuster M., Chen Z., Le Q. V, Norouzi, M., Macherey W., Krikun M., Cao Y., Gao Q., and Macherey K., (2016). "Google's neural machine translation system: Bridging the gap between human and machine translation", *arXiv preprint arXiv: 1609.08144*.

Zhong M., Wang D., Liu P., Qiu X. and Huang X., (2019). "A Closer Look at Data Bias in Neural Extractive Summarisation Models", in *Proc. of the 2nd Workshop on New Frontiers in Summarisation*.