DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2019

Evaluation of the Transformer Model for Abstractive Text Summarization

FREDRIK JONSSON

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Evaluation of the Transformer Model for Abstractive Text Summarization

FREDRIK JONSSON

Master in Computer Science
Date: August 9, 2019
Supervisor: Jonas Moll
Examiner: Olov Engwall
School of Electrical Engineering and Computer Science
Swedish title: Evaluering av Transformermodellen för abstraktiva textsammanfattningar

Abstract

Being able to generate summaries automatically could speed up the spread and retention of information and potentially increase productivity in several fields.

Using RNN-based encoder-decoder models with attention have been successful on a variety of language-related tasks such as automatic summarization but also in the field of machine translation. Lately, the Transformer model has been shown to outperform RNN-based models with attention in the related field of machine translation.

This study compares the Transformer model to a LSTM-based encoder-decoder model with attention on the task of abstractive summarization. Evaluation is done both automatically, using ROUGE score, as well as using human evaluators to estimate the grammar and readability of the generated summaries. The results show that the Transformer model produces better summaries both in terms of ROUGE score and when evaluated with human evaluators.

Sammanfattning

Att automatiskt kunna generera sammanfattningar ökar möjligheten att snabbt kunna sprida och ta del av information vilket potentiellt kan leda till produktivitetsökningar inom en mängd fält.

RNN-baserade enkoder-dekodermodeller med attention har visat sig vara effektiva inom många språkrelaterade områden såsom automatiskt genererade sammanfattningar men också inom exempelvis automatisk översättning. På senare tid har Transformermodellen överträffat RNN-baserade enkoder-dekodermodeller med attention inom det närliggande området automatiska översättningar. Denna uppsats jämför Transformermodellen med en LSTM-baserad enkoder-dekodermodell med attention både genom att använda det automatiska måttet ROUGE, men också genom att jämföra läsbarhet och grammatik i de automatgenererade sammanfattningarna med hjälp av mänskliga utvärderare.

Resultaten visar att Transformermodellen genererar bättre sammanfattningar både utvärderat med ROUGE och när de mänskliga utvärderarna används.

Contents

Acknowledgements

This thesis was conducted in cooperation with the company Findwise. I want to thank Findwise for providing their expertise, office space, as well as their support throughout the writing process.

A special thanks to my supervisors Jonas Moll at KTH as well as Johan Wikström at Findwise for providing me with invaluable feedback and pointers to help move the project forward.

I would also like to thank all of the people who took the time to answer the readability survey, a crucial part of the report.

Chapter 1

Introduction

1.1 Overview

The amount of information available on the internet and elsewhere in the form of written text has likely never been higher. A problem facing many is how to extract useful and relevant information from the sheer amount of data that can be accessed.

It is vital that the information can be easily searched but also presented in such a way that the user quickly can decide whether the information is relevant or not. One method that could potentially help with the second part is automated text summarization. Text summarization is used to extract only the most relevant information from a document or collection of documents, allowing the reader to faster absorb the key message from the text. Common use cases are abstracts in scientific papers, leads in newspaper articles as well as briefs within law. Humans usually do summarization by reading a text and paraphrasing paragraphs and sentences to compress the information as much as possible. This task is time-consuming and could benefit from automation, with research in the area going on at least since 1950s [1, 2]. There are two main overarching categories of automatic text summarization, extractive and abstractive [2]. Extractive techniques work by extracting written sentences from the source text, whereas abstractive methods rewrite sentences in source texts, which is more similar to how humans approach the same problem. Abstractive approaches could, in principle, generate summaries that are more efficient and natural sounding than extractive [3].

There is also a possibility that abstractive methods could provide insights into our understanding of language. When using abstractive summarization, the algorithm itself must ensure readability and grammatical correctness of the generated sentences, which requires the algorithm to extract knowledge about the language during training. For summarization, there is also the challenging task of recognizing what information to use for the summary and what to leave out. The approaches that have been used so far in the field of abstractive summarization has mainly relied on heuristics-based on linguistic and syntactic knowledge of language [4].

Lately, encoder-decoder models, usually in the form of two Recurrent Neural Networks (RNN), have proved to be good at interpreting and generating sequential data such as text. In this architecture, one encoder is used to read the input and encode it into a latent vector, whereas the decoder reads the data represented in this vector and generates a new sequence of text [5].

Due to the success of encoder-decoder models for text generation tasks a lot of research has been put into experimenting with different configurations of this architecture [6]. One approach has been to combine encoder-decoder models with an attention mechanism for tasks involving language generation such as machine translation and abstractive text summarization [7, 8, 3, 5]. The reason for introducing the attention mechanism was to improve interpretation and enable better generation of longer sequences of text. These models, that combine attention and the encoder-decoder architecture, are usually referred to as sequence2sequence (seq2seq) models and have played a big part in the increased interest in abstractive summarization [9, 10, 11]

Recently there has been an attempt at redesigning the RNN-based encoder-decoder model by relying more on the attention mechanism in the context of machine translation. A model developed by Vaswani et al. [12], called the Transformer, removed the need of the RNN part by combining the attention mechanism and feedforward layers. By doing so, they were able to reach state-of-the-art results, improving translation performance especially on longer sequences of text. The Transformer also proved to train faster than RNN-based seq2seq models, in part because it allows for more operations running in parallel during training [12].

The Transformer was initially developed and tested on the task of machine translation where it outperformed state of the art solutions by decreasing training time and increasing translation quality [12]. Due to the similarity of the two problems of machine translation and automatic summarization, it would be reasonable to expect that the Transformer would perform well on the abstractive summarization task as well.

1.2 Problem Formulation

Being able to generate summaries that are well written and reliably return relevant information has a potential to increase productivity in many fields. Especially fields that frequently rely on large amounts of written information such as within the fields of medicine or law. One common approach has been to use seq2seq models with attention for this task.

Recently the Transformer was able to outperform RNN-based seq2seq models for machine translation, especially due to its ability to run more operations in parallel as well as being better at handling long sequences. Comparing the Transformer to the RNN-based seq2seq model on the task of abstractive summarization would increase knowledge on how the removal of recurrence in neural networks affects summary quality.

Another aspect is the method of evaluation where most previous evaluations of seq2seq models on summarization tasks rely on automated metrics such as the ROUGE metric [13, 14, 10, 3, 4, 6]. The ROUGE metric has some limitations, mainly that it only looks at content by measuring n-gram overlap and not at the readability of the summary [15]. Combining the ROUGE metric with human evaluation could potentially provide a more comprehensive evaluation, allowing for both readability and grammar to be examined as well as content.

The main difference between RNN-based seq2seq models and the Transformer is the lack of recurrence in the latter. Since the Transformer is a relatively new model in comparison with recurrent models, there are still areas to explore, such as how the removal of recurrence affects learning and performance in different contexts.

The goal of this thesis is to investigate how well the Transformer performs on the abstractive summarization task in comparison with an RNN-based seq2seq baseline using both automatic and human evaluation methods. This knowledge would help with model selection for text summarization tasks and also provide insights into the readability of the output from encoder-decoder summarization models.

1.3 Research Question

This thesis will focus on two of the problem areas identified in section 1.2. Firstly, the lack of research in applying the Transformer model for text summarization and how it compares to a RNN-based seq2seq model with attention, secondly, the readability and quality of the automatically generated summaries as judged by human evaluators. The RNN-based seq2seq model with attention will hereon be referenced to as the seq2seq baseline.

The research questions explored are:

1. How does the Transformer model perform on the task of abstractive summarization in comparison with a seq2seq baseline, as measured with the ROUGE metric?

2. How does the Transformer model compare with a seq2seq baseline when evaluating using human evaluators to measure readability and grammar of generated summaries?

The expected result was that the Transformer would be able to reach similar or better results compared to the seq2seq baseline with shorter training time due to increased ability to run in parallel as well as a better ability to handle long input sequences. The ability to interpret and generate longer sequences could improve readability and grammar since dependencies between words within a sentence could be modeled better by the Transformer.

1.3.1 Limitations

The process entails preprocessing of data, model selection, hyperparameter selection and tuning as well as a final evaluation step for comparing the models.

Even though there are many available evaluation metrics, this thesis will focus on using ROUGE-1, ROUGE-2, and ROUGE-L F1 scores since these are commonly used which could allow for better comparison with previous results.

The human evaluation will only be conducted on the readability and grammatical aspect of the generated summaries and not on how well the generated summary reflects the original content.

Apart from using neural networks for abstractive summarization there has been a success in applying reinforcement learning in combination with neural networks as well as using combinations of extractive and abstractive techniques which have achieved high ROUGE scores [16, 17, 7]. This thesis project will focus on using neural network-based encoder-decoder architectures and will, therefore, exclude other methods from the literature review, as well as when comparing the final results to other studies.

Chapter 2

Background

The background chapter covers the theoretical background in the field of abstractive text summarization. The chapter presents the main theoretical concepts and explains the models used in this report in depth and introduces the reader to previous research in the field of abstractive text summarization.

2.1 Theory

2.1.1 Abstractive Summarization

As previously mentioned, there are two main approaches to automatic text summarization, abstractive, and extractive. The main difference between them is how information is extracted from the document and how the summary is generated. Extractive text summarization work on the principle of extracting the most relevant sentences or paragraphs in the source document.

Abstractive summarization, on the other hand, tries to rewrite and reformulate text from the original document, which is more similar to how a human does summarization.

Due to the difficulty of both extracting relevant information from a document as well as automatically generating coherent text, abstractive summarization has been considered a more complex problem than the extractive counterpart. This fact has also resulted in less research in the area of abstractive summarization in comparison to extractive [4].

The earliest approaches in abstractive summarization relied on statistical methods inspired by machine translation and usually focused on summarizing one sentence at a time [4]. Many of these first approaches were heavily reliant on heuristics and dictionaries for substitution of words [18].

One of the earliest approaches at implementing neural methods for end-to-end abstractive summarization was done by Rush, Chopra, and Weston who implemented an encoder-decoder model with attention. This model was based on earlier work in the field of machine translation [4].

2.1.2 Evaluation Methods

Human evaluators

Using human evaluators has so far been considered a qualitatively better way to evaluate since there are many aspects of summarization that are difficult to measure objectively by automated methods. When using human evaluators, some of the metrics commonly considered are coherence, conciseness, readability, and content [15]. Another approach is to ask evaluators to chose between two summaries, either two system-generated or a system-generated and a reference summary, and specify which one they prefer [11, 15].

Using humans has drawbacks in the form of time required and high costs [15]. Humans also lack in consistency as compared to automatic evaluators. There are examples of where the same judge scores the same summary differently when presented with the same text at two different occasions [19]. The issue of consistency in human evaluators is one of the arguments for using automatic summarization metrics for evaluation [15, 20]. Most automated metrics are constructed as to correlate to scoring done by human evaluators and do so reasonably well [19].

ROUGE

One of the automatic evaluation metrics is ROUGE score, which is a measure of overlapping n-grams in the generated summary and one or several reference summaries constructed by humans [15]. ROUGE-2 will, for example, measure overlap of 2-grams between a system generated summary and a reference summary [15]. The most commonly used versions in previous studies are ROUGE-1, ROUGE-2, and ROUGE-L, where ROUGE-L is the measure of the longest common sub-sequence. ROUGE is a popular metric in part since it has been shown to correlate with human judgments of summary quality [19]. Since many papers use ROUGE when evaluating abstractive summarization models, the metric is suitable for comparing models to each other [8, 21, 3, 4, 6].

METEOR-metric

METEOR is another automatic evaluation metric used for evaluating automatically generated summaries [3]. METEOR was initially developed for automatic evaluations in machine translation as it calculates sentence level similarity scores between two sentences, usually a generated translation and a reference translation [22]. In a case where a reference summary exists, the METEOR metric could just as well be used to measure the similarity between this reference summary and a system generated summary.

2.1.3 RNN encoder-decoder

The encoder-decoder model is extensively used for tasks that map one sequence to another sequence [10]. Since language is a sequence of letters, words, sentences, etc., the model is used in many language-related tasks such as machine translation and summarization. The basic encoder-decoder model for language tasks in Figure 2.1 usually consists of two RNN where one RNN works as the encoder and the other one as a decoder [23].

The encoder is given a sequence of vectors as an input $\mathbf{x} = \{x_1, ..., x_n\}$. Each vector is usually processed into a hidden state h_t of the encoder. When using a standard RNN, each vector is processed by calculating [5]:

$$h_t = f(x_t, h_{t-1}) \tag{2.1}$$

In equation 2.1, f could be a combination of an activation function and a weight matrix multiplication of the inputs but is usually based upon Gated Recurrent Units (GRU) or Long short-term memory (LSTM) cells [9]. The difference from a feedforward network is the inclusion of the hidden state h_{t-1} from the previous time step in the input to f. The resulting encoding in the last time step is often referred to as a context vector c [5].

The context vector c is later part of the information that is fed to the decoder. The decoder also has its hidden states s_t that work by the same principle as the encoder, calculated as [23]:

$$s_t = f(s_{t-1}, c, y_{t-1}) \tag{2.2}$$

The final output is then calculated as:

$$y_t = g(s_t, c, y_{t-1}) \tag{2.3}$$

where g is an activation function producing valid probabilities e.g. softmax [23].

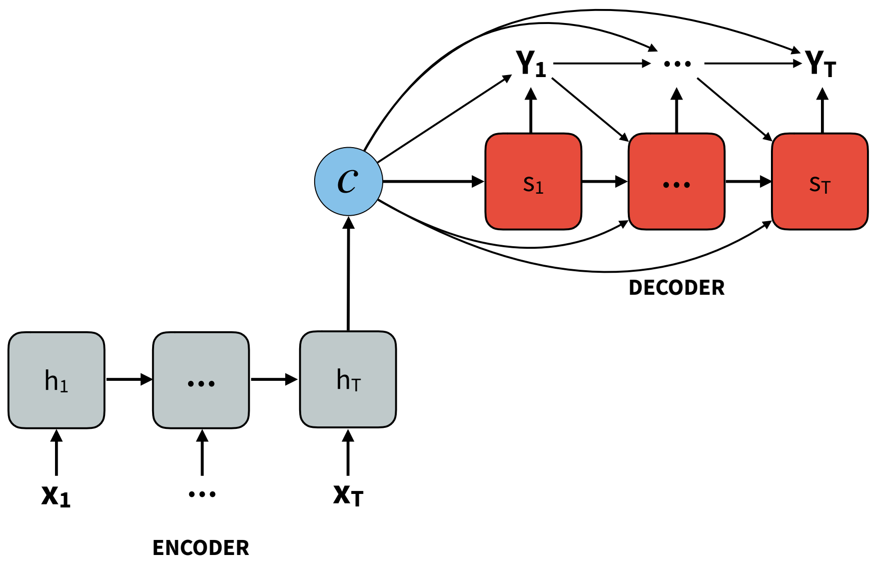


Figure 2.1: Encoder Decoder model

2.1.4 Attention

The attention mechanism was introduced by Bahdanau, Cho, and Bengio [5] in the context of machine translation. A bottleneck identified in earlier encoder-decoder RNNs was the limitation of encoding the entire input sequence into one fixed context vector [5]. By implementing the attention mechanism, the quality of the translation could be improved, mainly when translating longer sequences of text [5].

On a high level, the attention mechanism adds the functionality of storing all of the hidden states of the encoder, see figure 2.2. The decoder is then able to use the information stored in all of these hidden states as an input when calculating the next word in the output sequence.

More specifically, instead of calculating a single common context vector c, a specific context vector for each decoding step is calculated as a weighted sum of all the previous hidden states $(h_1, ..., h_T)$ as in equation 2.4 [5].

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j \tag{2.4}$$

The weights in matrix α_{ij} is calculated as:

$$a_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T} exp(e_{ik})} \tag{2.5}$$

Where e_{ij} is an alignment model that learns to match the encoder hidden states to the decoder hidden states [5]. The alignment model could be a feed-forward neural network which is given the concatenation of the decoder and encoder hidden states [5]. Other suggestions are made by Luong, Pham, and Manning, for example dot product attention, in which the dot product between the decoder hidden state and the decoder states is used as the alignment model [24]. By giving more weight to relevant words for each decoding time step longer sequences can be remembered, and only the most relevant information is fed to the decoder [5].

A common addition to the standard seq2seq model is to use bidirectional encoders [5, 3, 10], meaning that the input, as well as the input in reverse are encoded into hidden states. Both the forward and the backwards hidden states are then concatenated and fed to the decoder. Bidirectional encoders have been shown to improve performance when encoding longer sequences [5].

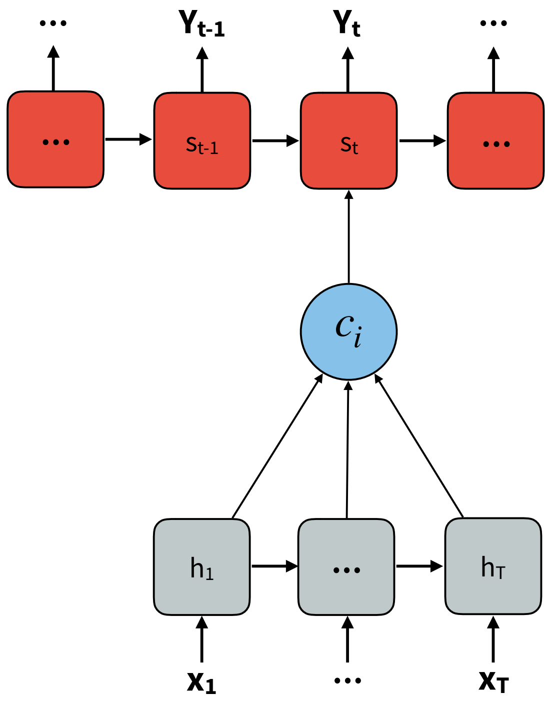


Figure 2.2: Encoder Decoder with attention

2.1.5 Transformer

In the encoder-decoder model with attention the previous hidden state h_{t-1} is required to calculate the current hidden state h_t [12]. This sequential architecture limits efficiency during training, as it is not fully utilizing the power of Graphics Processing Units (GPU). As a solution to this problem the Transformer model was developed by Vaswani et al. [12]. In this model, recurrence is removed entirely and only the attention mechanism in combination with feedforward networks is used for mapping dependencies between the input and the output [12]. The attention in the Transformer is calculated using Scaled Dot-Product Attention, which is based upon the Luong dot product attention with an additional scaling factor [12]. It calculates the dot product between a query, a key and a value matrix Q, K and V see equation 2.6.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.6}$$

The $\sqrt{d_k}$ term is used for scaling where d_k is the dimension of the input K. What the K, V, and Q matrices represents depends on the context. In the encoder and decoder, they are all outputs from the previous layers. In this context, the attention mechanism is called self-attention [12]. In encoder-decoder layers the Q matrix will be the output from the previous decoder layer and K and V will be output of the encoder layer [12]. In this context, the attention mechanism works similarly to the attention mechanism in the RNN-based encoder-decoder models [12].

The Scaled Dot-Product mechanism is run in parallel on several linear projections of the input, the output is later concatenated and multiplied by an additional weight matrix. This implementation is called Multi-Head Attention by the authors. The output of the Multi-Head Attention layer is later used as input to a feedforward layer. Additionally residual connections are utilized to bypass the Multi-Head Attention and feedforward layers. The residual connections are added to the output of these layers followed by normalization. [12].

Both the encoder and decoder have the same structure apart from an additional version of the Multi-Head Attention in the decoder called Masked Multi-Head Attention. The Masked Multi-Head Attention masks future positions in the output preventing the decoder from "peeking" into the future during training [12].

Since the Transformer does not utilize recurrence, the authors had to add a positional encoding to the input so that the model could keep track of the order of the input sequence. In the original article this was done using a sine and cosine positional encoding, but other forms of positional encoding could be used as well [12].

An illustration of the Transformer architecture can be seen in figure 2.3. The input to this model will be positional encoded word embeddings and the final decoder output will be run through a linear followed by a softmax layer to convert the outputs to probabilities [12].

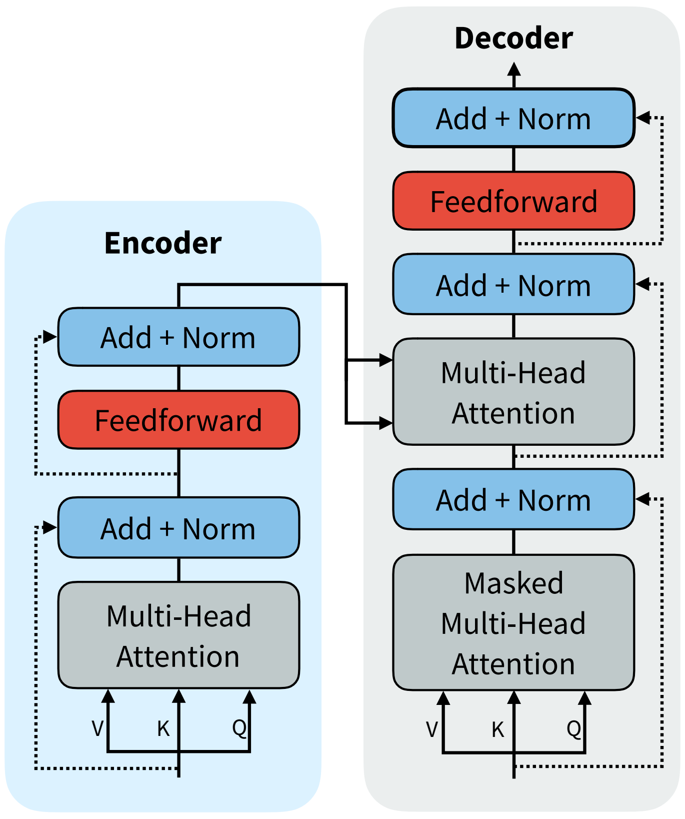


Figure 2.3: Transformer

2.2 Previous Work

A typical modern use case for neural networks is language modeling where the networks are used to predict the next word in a sequence based upon the previously seen words. Language modeling with neural networks has been done at least since the 1990s [25].

In the field of machine translation, neural networks were commonly used in this fashion, often to provide input for other non-neural network-based translation systems to increase their performance [9]. Lately, neural networks have been used successfully in an end to end fashion for directly translating sequences of words between languages called Neural Machine Translation (NMT). One of the first attempts at this was done by Sutskever, Vinyals, and Le in 2014 [9]. They applied an LSTM-based encoder-decoder model that could handle varying length input and output sequences by encoding one sequence at a time into a fixed size hidden vector which later was decoded sequentially [9].

Another early implementation of these types of encoder-decoder models for machine translation was done by Bahdanau, Cho, and Bengio in 2014 [5]. They combined the basic encoder-decoder with the attention mechanism, which was shown to increase translation performance on long sequences.

This attention mechanism was later adapted by Rush, Chopra, and Weston in 2015 in the context of abstractive summarization [4]. They use the attention mechanism as described by Bahdanau, Cho, and Bengio but make changes in the general model by implementing a convolutional encoder and implementing beam search in the decoder. When using beam search, several of the most likely words are kept at each time-step, allowing the decoder to generate many possible output sequences and choose the most likely one. This model was applied on single sentence summaries with good results, improving on several strong baselines when measured by ROUGE score. The study makes no use of human evaluators for scoring but compares the model output to summaries generated by humans.

Another implementation of the Bahdanau, Cho, and Bengio [5] attention mechanism in the context of text summarization was done by Nallapati et al. in 2016 [10]. With only small changes to the original encoder-decoder architecture, they were able to reach state of the art performance on abstractive text summarization tasks. Apart from implementing the new model, they also introduce a new data set for text summarization, the CNN/Daily Mail corpus. For evaluation, they utilized the ROUGE metric, but there is no use of human evaluators.

Two of the most common issues with abstractive summarization that could lead to unnaturally sounding summaries are words in the input that are not part of the vocabulary, so called out of vocabulary words, as well as unnecessary word repetition in the generated output sequence. These issues are approached by See, Liu, and Manning in 2017 [3] in the context of multi-sentence summarization using the seq2seq model. To counter the out of vocabulary problem, they implement a pointer-generator network that allows for copying of words from the input sequence as well as generating words from the pre-defined vocabulary [3]. In response to the word repetition in the output, they implemented a coverage mechanism in which a coverage vector is used as input to the attention mechanism. The coverage vector keeps track of states that have been visited previously and thus ensures that the attention mechanism does not generate repetition in the output [3]. This approach was shown to be able to reduce the amount of repetition in the output as measured by the fraction of duplicate n-grams [3]. The model is evaluated by using ROUGE, METEOR, and by measuring the number of duplicate n-grams.

A recent attempt to redesign the seq2seq-attention model for machine translation was done by Vaswani et al. in 2017 [12]. They developed the Transformer model which, is described in subsection 2.1.5 above. In their approach, the RNN parts are removed entirely making the model rely only on attention to learn the dependencies between input and output sequences [12]. When applying the Transformer on a translation task, it reaches state of the art performance requiring less training time in comparison to encoder-decoder models with attention [12]. The transformer model has since been used in language modeling for a variety of tasks achieving state of the art results [26].

A recent implementation of the Transformer for text summarization was done by Liu et al. in 2018 [11]. Here the Transformer is applied on abstractive summarization in combination with extractive techniques for generating Wikipedia articles. In their article [11], they evaluate the Transformer, a seq2seq model, and a modified version of the Transformer that only uses the decoder part. The evaluation showed that both of the Transformer variants outperformed the RNN-based seq2seq model in terms of ROUGE score and when using five human evaluators. Since this evaluation was conducted on an entirely new data set, using Wikipedia references, comparing the results to other attempts at abstractive summarization is difficult.

A summary of some of the previous results on the CNN/Dailymail data set using encoder-decoder models can be seen in table 2.1.

Table 2.1: Summary of some of the previous results using ROUGE-F1 score on the CNN/Dailymail data set using encoder-decoder models for abstractive summarization. Nallapati et al. [10] uses a max input length of 800 words and max output length 100 words on the anonymized data set. See, Liu, and Manning [3] uses max input of 400 words and max output of 120 words. The lead 3 baseline uses the first three sentences from the original text as a summary.

Model Name	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**Author**
lead-3 baseline	40.34	17.70	36.57	[3]
pointer-generator + coverage	39.53	17.28	36.38	[3]
words-lvt2k-temp-att	35.46	13.30	32.62	[10]
seq-to-seq + attn baseline (50k vocab)	31.33	11.81	28.83	[3]

2.2.1 Data sets

For training an abstractive neural network based summarization algorithm, a common approach is to use pairs of texts and human written summaries. One such data set, based on CNN and Dailymail articles, was developed for this task by Nallapati et al. [10]. The CNN/Daily Mail data set consists of more than 300,000 human-written text summary pairs [10].

Another example of a data set that has been used in previous literature is the Gigaword corpus, also based upon news articles [13]. There also exists a more detailed annotated version of this corpus, constructed by Napoles, Gormley, and Van Durme [27]. This data set is much larger than the CNN/Dailymail data but uses headlines and/or the first sentence in the text as reference summaries, which are often shorter in length than in the CNN/Dailymail counterpart [13].

2.2.2 Knowledge Gap

Seq2seq models using recurrence have performed well on both abstractive summarization and on neural machine translation but have recently been surpassed by the Transformer model in the field of machine translation. Evaluation of the Transformer for abstractive summarization and how it performs in comparison with encoder-decoder models with attention is yet to be fully explored. Another aspect to be noted from the previous research is the lack of systematic investigation, using human evaluators, into the readability of summaries generated by RNN-based seq2seq models with attention.

Chapter 3

Method

This chapter describes the preprocessing steps and the methods for evaluating the Transformer on the abstractive summarization task. A description of the specific frameworks and parameters used for training both the Transformer and the seq2seq baseline is presented in this section as well.

3.1 Tensor2tensor-library

Both the Transformer and the reference model were implemented using the tensor2tensor library [12]. The tensor2tensor library was chosen since it is the official library cited in the paper by Vaswani et al. [12] presenting the Transformer.

This library provides methods for preprocessing, training, and evaluation on the CNN/Dailymail data, as well as several predefined model architectures that could be used for training on the data set. The library expects the user to define what type of input data to expect, where the data is located and where to store checkpoints. Hyperparameters as well as other options for model architecture such as number of layers and layer size could also be specified. All of these tools helped to simplify preprocessing and model implementation which allowed more time to be spent on training, evaluation and analysis of the results.

Hyperparameter selection is also facilitated as there are default hyperparameter sets provided for each model that could be used as a starting point.

The versions used for training and evaluation are tensor2tensor 1.13.2, Tensorflow 1.13.1, and Python 3.6.

3.2 Preprocessing

The CNN/Dailymail data set consists of 311,971 document summary pairs from newspaper articles. Splitting into train, test and eval sets can be done automatically when generating the data set using the tensor2tensor library. The default setting was used, resulting in the final train-test-evaluation split as follows:

- Total samples: 311,971 (100%)
- Number of train samples: 287,113 (92.0%)
- Number of evaluation samples: 13,368 (4.3%)
- Number of test samples: 11,490 (3.7%)

This split has been used in previous work [10, 3], which was a motivation for using it also in this thesis. The version of the CNN/Dailymail data set used in this paper is based on the non-anonymized version collected from See [28].

The input was tokenized to sub-words into a vocabulary size of 32,781 sub-words. Using sub-words instead of full-length words as tokens allow for smaller vocabulary sizes, as well as allowing the decoder to generate words that it has not seen during training and has been shown to increase performance on machine translation tasks [29].

When training the neural networks, the maximum size of the input will need to be known beforehand. Since the length of the texts varies, there are two main ways to handle this. During training, the maximum input length (number of sub-words) could be controlled using the *max_input_length* flag in the tensor2tensor library. This setting would pad shorter sequences up to this max length and drop sequences that are longer, There is also an option to truncate the max input and output lengths using the settings *max_input_seq_length* and *max_output_seq_length*. Truncating and or limiting the input length will reduce memory requirements and potentially speed up training.

Previous studies suggest that seq2seq models, as well as the Transformer, have some difficulties with input lengths exceeding ~500 tokens [11]. Limiting the *max_length* to 500 would however reduce the size of the data set, which also could have a detrimental effect on training since fewer examples are available for the models. Nallapati et al. [10] uses a max length of 800 words during the training of their seq2seq model on the CNN/Dailymail data set. Others have truncated the input sequences to a maximum of 400 tokens and output to 120 tokens which has been shown to increase performance [3].

In this thesis, sequences longer than 2048 were dropped since this only reduces the data set by ~1.4‰. The input was later truncated to 512 sub-words, and the output was truncated to 128 sub-words. Using this truncation results in a data set similar to what was used by [3]. Using 512 as max input length when using sub-words roughly corresponds to an input size of 400 when using full-length words. It was also expected to be better to use an input length that is a multiple of 8 as this is preferred when training on Tensor Processing Units (TPU) [30]. Initial testing showed a substantial reduction in training time when truncating with no apparent performance decrease in terms of ROUGE score.

As can be seen from figure 3.1, almost all of the data is captured by input lengths of less than 2048 sub-words and targets of less than 128 sub-words.

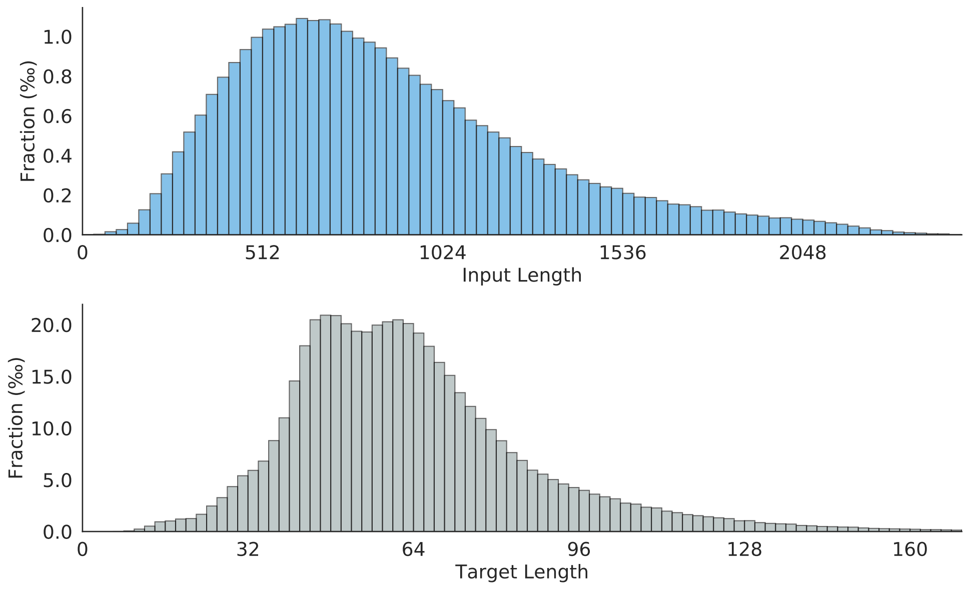


Figure 3.1: Histogram over input and target lengths in the CNN/Dailymail data set.

3.3 Transformer

The Transformer is implemented from the tensor2tensor library using the Transformer model. This model has several default hyperparameter sets that could be used as a starting point for a variety of machine learning tasks. The hyperparameter sets experimented with in this thesis were: *transformer_prepend*, *transformer_base_v1*, *transformer_base_v3* and *transformer_tpu*.

For summarization on the CNN/Dailymail data set the hyperparameter set *transformer_prepend* is recommended by the authors of the tensor2tensor library. The set *transformer_base_v3* is the default and is also referred to as *transformer_base*. The setup *transformer_base_v1* has been used in previous work to generate summaries with high ROUGE scores [11].

A comparison of some of the most significant differences between the hyperparameters can be observed in table 3.1.

Table 3.1: Hyperparameter sets for the Transformer.

Hyperparameter	**tpu**	**prepend**	**base_v1**	**base_v3**
optimizer_adam_beta2	0.997	0.98	0.98	0.997
relu_dropout	0.1	0.1	0	0.1
attention_dropout	0.1	0.1	0	0.1
learning_rate	0.2	0.2	0.1	0.2
layer_postprocess_sequence	da	da	dan	da
layer_preprocess_sequence	n	n	none	n
optimizer	Adafactor	adam	adam	adam
learning_rate_constant	2	1	1	2
learning_rate_warmup_steps	10000	8000	4000	8000

3.3.1 Hyper parameters

Optimizer

The default optimizer used in the tensor2tensor library for most models is the Adam optimizer [31]. The Adam optimizer has three adjustable values, α, β_1, β_2 where α is the step size and β_1 and β_2 are the first and second moment decay rates.

For the *transformer_tpu* the optimizer Adafactor is used, this optimizer is based upon Adam but requires less memory [32].

Layer pre- and postprocess

The *layer_preprocess_sequence* and *layer_postprocess_sequence* settings specify which operations to perform on the input and output of each layer in the Transformer. The settings are "d" for dropout, "n" for normalization and "a" for residual connections. The "a" option is only used in the post-process. Using "dan" is the default setting used in the original report by [12], which means using dropout followed by adding layer input to layer output as well as normalizing the result.

Dropout

Dropout is a regularisation method used to prevent over-fitting by deactivating units in a neural network [33]. Dropout is set to a value between 0 and 1, which specifies the probability that a unit will be dropped. Dropout can be specified to different values on different layers. During training the default value is selected and during inference this value is set to 0.

Warmup

Warmup is a method for starting training with a lower learning rate and then increasing it after a certain number of steps [34]. Some options are to set a constant initial learning rate or to increase it linearly up until the selected number of warmup steps. The number of warmup steps are specified by the *learning_rate_warmup* steps hyperparameter and is set to the default value for each model.

There is also a learning rate decay that performs the opposite, meaning that after the warmup phase, the learning rate starts to decrease again according to the selected decay scheme [35].

Model Size

Apart from the hyperparameters specified above there is also the option to change the number of layers and attention heads. This can be done by selecting one of the several predefined versions of the Transformer in the tensor2tensor library or by changing the *num_heads* and *num_hidden* layers parameters.

In this thesis the default version is selected which uses six encoder and decoder layers, each with eight Multi-Head Attention layers running in parallel.

3.4 Seq2seq Baseline

The seq2seq baseline is an LSTM based seq2seq model also implemented from the tensor2tensor library. The default implementation uses two hidden layers of size 128. There are two different attention mechanisms to choose from Luong and Bahdanau attention, as well as the option to use bidirectional encoders.

3.5 Hardware

Model training and evaluation has been done using Google Cloud TPUs. As of now, two versions exist TPU v2 and v3. The TPU v2 is freely available when using Google Colab Notebooks, which helps to keep costs down. The TPUs are optimized for machine learning using Tensorflow and reduces training time and cost of training machine learning models compared to GPUs. Both TPU versions have four independent chips containing two cores, where the TPU v2 has 8 GB associated memory for each core [36].

For storage of the data set and model checkpoints Google Cloud Storage was used as this was required for training on TPUs and as Google Colab does not provide persistent storage.

3.6 Evaluation

The automatic evaluation was done using ROUGE-1, ROUGE-2, and ROUGE-L F1 scores. ROUGE score is used partly as it has been shown to correlate with human evaluation of summary quality [19], but also because it is a commonly used metric that allows comparison to previous studies. All the ROUGE variants can be used either to measure recall, precision or, by using F1 score which combines recall and precision. In this report the F1 score was chosen as it is not affected as much by summary length and since it also provides a balance between recall and precision [10].

The tensor2tensor library could be set to calculate ROUGE score during training but, there were two issues with using this approach. Firstly there is an issue when using TPUs which makes the training freeze when restoring a checkpoint after evaluation [37]. Secondly, the ROUGE scores reported are calculated differently than the ROUGE scores produced by the official ROUGE-1.5.5 implementation, which is used in previous studies [10, 3].

Instead the py-rouge script [38], based on the official ROUGE-1.1.5, implementation was used to calculate ROUGE score. For this to work the model- and the reference summary had to be processed so as to contain one sentence per line, as this is expected by the py-rouge script.

The py-rouge implementation reports a confidence interval for the calculated ROUGE values, which are presented in conjunction with the results. The evaluation was done on different input lengths, but at all times the output was truncated to 128 sub-words.

3.7 Inital Testing

For initial testing, the default hyperparameter settings were used in most cases except for the *batch_size* which was set to 4096 and the change in input and output sequence length as described above. The seq2seq baseline was evaluated both with and without a bidirectional encoder. Another change was the hidden size for the seq2seq baseline, which was tested with a size of both 128 and 256.

The hyperparameter sets were initially evaluated to 100,000 steps to compare their performance; the results can be seen in chapter 4. The hyperparameter set yielding the best results on the evaluation set in terms of ROUGE-2 F1-scores was selected for further training.

For both the Transformer and the reference model, the evaluation was run on the same 100 randomly sampled texts from the evaluation set. Decoding was performed using *beam_search* with a *beam_size* of 4 and *alpha* of 0.6. The score was then calculated on the generated summary by comparing it to the reference summary using the py-rouge script.

Beam size of 4 means keeping the 4 most likely predictions in each time step and expanding upon each of these to find the most likely final summary. A beam size of 1 would only keep the most likely subword at each time step, what is known as greedy decoding. The alpha parameter is used for normalizing the length of the generated summaries as to mitigate the effect of sentence length on the final likelihood. As the final likelihood is calculated by multiplying the probabilities of all the generated words, long sequences are disadvantaged as the final likelihood would be a very low number. By lowering alpha the penalty of generating long sequences will be reduced.

3.8 Comparing the Transformer to the seq2seq baseline

A common stop criterion is to train models until they start over-fitting. Many systems of this type do not tend to over-fit, even after a long time of training making this stopping criterion unfavorable [35]. The models were instead trained to 500,000 steps using the same settings as during initial evaluation. This approach has drawbacks as it does not guarantee that the models have reached their maximum score, but this was considered a reasonable stopping criterion after observing the low rate of performance increase after about 100,000 steps.

The best performing model on the evaluation set in terms of ROUGE-2 score was chosen for comparison on the test set and by using human evaluators. When comparing the best models, both were given the same 1000 randomly sampled texts from the test set with *beam size* 10 and *alpha* 0.6. The score was then calculated on the generated summaries by comparing them to the reference summaries using the py-rouge script. Models were tested and compared on different input lengths to see how this would affect the quality of the summaries.

3.9 Human evaluation

For the human evaluation, five linguistic quality markers developed by DUC 2005 [39], were used. These metrics are also used by Liu et al. [11] when evaluating text summaries and are explained as follows.

- Grammatical: No datelines, capitalization errors, or ungrammatical sentences.
- Non-redundancy: No unnecessary repetition.
- Referential clarity: Easy to identify references of pronouns and nouns.
- Focus: Sentences should only contain information that is related to the rest of the summary.
- Structure and coherence: The summary should be well-structured and organized.

Ten documents were chosen from the test set to be summarized, using the same settings as in the automatic evaluation for decoding. Each category was ranked from 1-5 where: 1 = very poor, 2 = poor, 3 = barely acceptable, 4 = good and 5 = very good. This evaluation looks at the readability and grammar of each summary but does not evaluate content. By using this approach, there is no need to show the original text to the participants during the evaluation, reducing the size of the survey. The evaluation was distributed using a web form which could be sent to the participants. The participants were mainly students and teachers at KTH, employees at Findwise, and teachers at an upper secondary school in Sweden. The exact identity of each participant is not known as the survey was anonymous, but the recipients were chosen to be people with an ongoing or completed college or university degree as this was considered to provide sufficient background knowledge in English for the evaluation. The layout of the form which was constructed using Google Forms can be seen in figure A.2 of the appendix. A total of 15 people responded to the survey.

Chapter 4

Results

This chapter presents the results from the initial testing as well as the final evaluation using both ROUGE score and human evaluations. The initial test runs were done to compare the different default hyperparameter sets in the tensor2tensor library as well as to examine the impact of varying the max input length and the effect of using different hyperparameter settings.

4.1 Results from Initial Testing

4.1.1 Results from Initial Testing: Transformer

The results from the initial evaluation of the Transformer up to 100,000 steps can be seen in figure 4.1. The *transformer_prepend* option initially seemed to perform best, but after inspection of the outputs, it was shown that the model was copying the input text. By copying the input high ROUGE scores could be achieved if truncating the outputs but as the intention of the study was to look at abstractive summarizations, the *transformer_prepend* set was excluded from further training. The *transformer_base_v1* was not able to learn at all and stops improving at around 1-2 in ROUGE-2 score. The hyperparameter *transformer_base* seemed to learn to produce summaries, but the increase in ROUGE score is slower than the hyperparameter set *transformer_tpu*.

The hyperparameter set *transformer_tpu* was chosen for further evaluation as it reached the highest ROUGE-2 score as well as having the fastest training time.

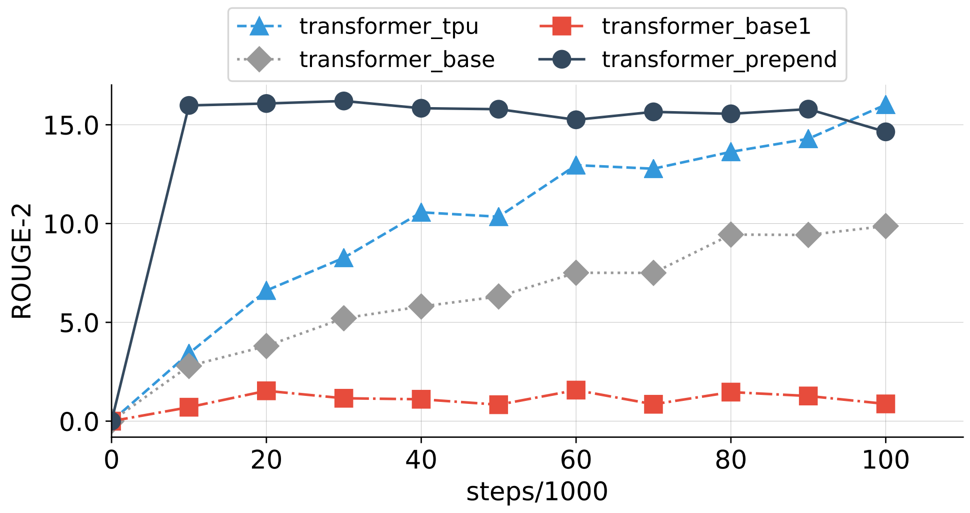


Figure 4.1: ROUGE-2 F1-score from initial testing on Transformer up to 100,000 steps.

4.1.2 Results from Initial Testing: seq2seq baselines

The results from the initial evaluation of the seq2seq baselines can be seen in figure 4.2.

Using bidirectional encoders as well as increasing hidden size increased training time, but also seemed to improve performance. Luong and bahdanau attention performs similarly, but *luong_attention* requires less memory than *bahdanau_attention*.

Since the best results on the initial evaluation were achieved with the *luong_attention*, a bidirectional encoder and a hidden size of 256, this model was chosen for further training.

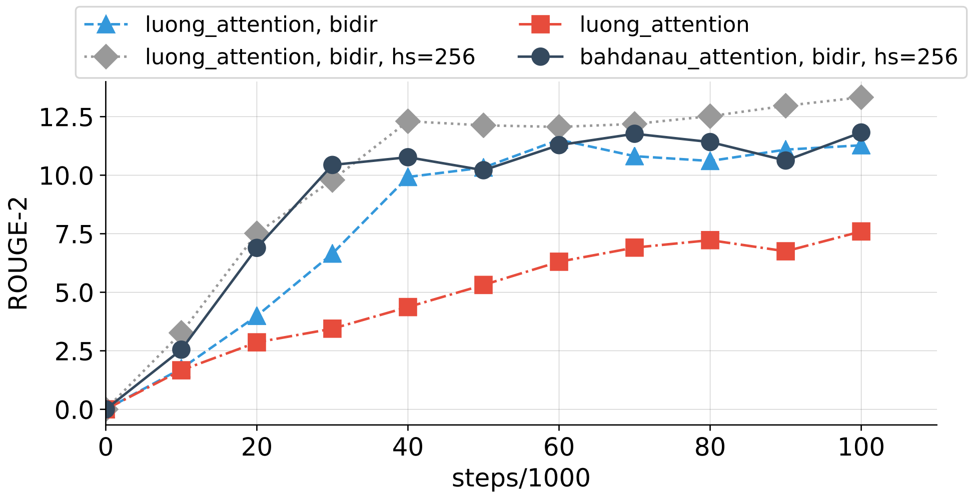


Figure 4.2: ROUGE-2 F1-score from initial testing on seq2seq baselines up to 100,000 steps.

4.2 Comparison of training time

Since the models are trained up until a set number of training steps a comparison is also made on how fast the models train. A summary of total training time up to 100,000 steps can be seen in table 4.1. From the summary it can be seen that the Transformer generally trains faster than the RNN-models. It can also be observed how the addition of bidirectional encoders and increasing the hidden size is affecting training time.

Table 4.1: A comparison of the training times, in hours, up to 100,000 steps for all the tested models. Models are sorted by training time from low to high.

Model	**Hours to 100,000 steps**
transformer_tpu	1.29
transformer_base1	1.46
transformer_base	1.52
transformer_prepend	1.52
luong_attention	1.57
luong_attention, bidir	2.79
luong_attention, bidir, hs=256	5.70
bahdanau_attention, bidir, hs=256	5.88

4.3 Comparison of Transformer and the seq2seq baseline

Table 4.2 shows ROUGE-1, ROUGE-2 and ROUGE-L F1-score for the models achieving the highest scores during training, up to 500,000 steps, which in this case was *transformer_tpu* and *luong_attention* with a bidirectional encoder and a hidden size of 256.

Both of the models were evaluated on 1000 randomly selected samples from the test set on different input lengths. The Transformer model outperforms the seq2seq baseline in all categories for all input lengths. The size of the 95 % confidence interval as reported by the py-rouge script is at most $\pm$ 0.75 ROUGE score, for all ROUGE variants, for both of the models. The Transformer achieves a ROUGE score that is consistently between ~28-45% higher than the seq2seq baseline.

Table 4.2: Mean ROUGE F1 scores for best Transformer and seq2seq baseline for different max input lengths. The size of 95% confidence interval for reported ROUGE scores are at most $\pm$ 0.75.

max input length	**Model**	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**
1024	transformer_tpu	39.06	16.76	35.93
	luong_attention, bidir, hs=256	28.63	11.56	26.16
512	transformer_tpu	39.22	16.93	36.00
	luong_attention, bidir, hs=256	29.73	12.39	27.15
256	transformer_tpu	39.15	16.89	35.99
	luong_attention, bidir, hs=256	30.47	12.74	27.79

4.4 Human Evaluation

The results from the human evaluation, showing mean scores and standard deviations can be seen in table 4.3. The mean is calculated across all raters and all documents for each model and each category. The difference in the score was also evaluated using a Welch Two Sample t-test with p = 0.001, showing that all differences are statistically significant.

Table 4.3: Mean scores and standard deviations for human evaluation in the categories G=Grammatical, NR=Non-Redundancy, RC=Referential Clarity, F=Focus, SC=Structure and Coherence

	transformer_tpu		**luong_attention, bidir, hs=256**			
Category	M	SD	M	SD	p	t
G	3.63	± 1.13	3.17	± 1.13	5.01e-04	3.52
NR	4.02	± 0.99	2.05	± 1.24	6.41e-39	15.27
RC	3.40	± 1.29	2.84	± 1.21	1.34e-04	3.87
F	3.37	± 1.31	2.55	± 1.20	4.17e-08	5.63
SC	3.09	± 1.31	2.02	± 1.09	3.44e-13	7.63

Chapter 5

Discussion

This chapter discusses the results from the initial testing, the comparison of ROUGE score between the Transformer and the seq2seq baseline, as well as the human evaluation. Limitations of the methods used, future work and societal, ethical and sustainability aspects are also discussed.

5.1 Automatic Evaluation

5.1.1 Results from initial Testing

For the Transformer, the different hyperparameter sets perform very differently. The hyperparameter set *transformer_prepend* yields good ROUGE scores but is learning to copy its inputs rather than generating new summaries. If truncating the output, it would result in similar results as a lead-3 baseline that are used in some other papers [3, 7]. These lead-3 baselines perform in level with the best abstractive summarization models on the CNN/Dailymail data set as measured by ROUGE score.

The hyperparameter set *transformer_base_v1* does not perform well, which is contrary to the results by Liu et al. [11], this could be due to changes made in the default parameters in the tensor2tensor library since their implementation. The issue is likely the hyperparameters *layer_postprocess_sequence* and *layer_preprocess_sequence* as changing these to "da" and "n" respectively seems to have a positive impact on learning.

Both the *transformer_base* and the *transformer_tpu* seems to perform well, the main difference being that the *transformer_tpu* learns faster as it is optimized for the TPU. Another aspect is the use of the more efficient Adafactor optimizer, which requires less memory than the Adam optimizer used in the *transformer_base* [32].

For the seq2seq baselines, all of the hyperparameters tested perform very similarly. There is a noticeable increase in training time when using bidirectional encoders as well as when increasing the hidden size of the models, but this also seems to be able to achieve higher ROUGE scores. Increasing hidden size as well as using a bidirectional encoder would, in theory, increase the ability for a these types of models to learn on longer sequences, so it is not unexpected that these two combinations seemed to work the best [5].

5.1.2 Comparing the Transformer to the seq2seq baseline

When looking at the final results and comparing the Transformer to the seq2seq baseline, it can be seen that the Transformer performs about ~28-45% better on all of the ROUGE metrics tested. Both of the models have 95% confidence intervals for the ROUGE score within $\pm$ 0.75 of the mean, indicating that the Transformer produces summaries that are more similar to the reference summaries.

When comparing the results in this thesis project to previous studies, the Transformer performs close in terms of ROUGE score to the pointer-generator + coverage model developed by See, Liu, and Manning [3].

The ROUGE scores from the seq2seq baseline presented in this thesis report is similar, although slightly lower, than the results presented by See, Liu, and Manning [3] on their RNN-based seq2seq model with attention. Both the Transformer and the seq2seq baseline perform worse in terms of ROUGE score than the lead-3 baseline reported by [3].

An exact comparison of the ROUGE scores cannot be made due to small differences in input and output lengths as well as the use of the anonymized data set by [10]. Another difference is that the models in this thesis project are evaluated on a subset of 1000 random samples from the test set, whereas both [10] and [3] seem to use the entire test set when evaluating.

5.2 Human Evaluation

Even though the number of survey respondents and number of articles evaluated is quite low, the differences between the model score are significant in all of the categories. The Transformer model achieves higher mean scores in all five categories, which correspond with the results from ROUGE evaluation. The most significant difference in mean score was observed in the category non-redundancy, which corresponds with observations and with previous studies suggesting that the seq2seq baseline tends to repeat itself. Looking at the overall results, the average score for the Transformer is somewhere between barely acceptable and good for most of the categories except for non-redundancy, where it is between good and very good. For the seq2seq baseline, most of the results are between poor and barely acceptable except for referential clarity and grammatical where it reaches slightly above barely acceptable. The score from the human evaluation shows that even if the ROUGE score is high for the Transformer, there are still improvements to be made in terms of readability and grammar.

Another observation from looking at the generated summaries is that the Transformer model tends to generate shorter sequences than the seq2seq baseline, which also could affect scoring.

5.2.1 The Importance of Truncation Length

How the input and output length is truncated affects training time but also score during evaluation. In this thesis project, a similar truncation of the input as in [3] was used, 512 sub-words compared to 400 words. Another common approach has been to truncate the output to 75 bytes before evaluating using ROUGE as this has been a standard set in previous Document Understanding (DUC) conferences [10].

When comparing the Transformer against the seq2seq baseline, it was noticed that truncating output length improved scores for the seq2seq baseline while at the same time slightly lowering the scores for the Transformer model. The reason for this is not investigated in depth but could be related to the issue with repetition in the output of the seq2seq model. Truncating will reduce repetition which could have an effect on ROUGE precision if the repeated output is not part of the reference summary.

5.3 Limitations of Study

The repetition in output when using RNN-based seq2seq models is a known problem and has mostly been solved by using a coverage vector that keeps track of words that have already been generated from the source sequence. Coverage has not been implemented in this thesis and would probably have increased performance for the seq2seq baseline. When doing the human evaluation, the categories are supposed to be evaluated separately, where word repetition has its own category. There is, however, a risk that the word repetition affects the results in other categories as well.

Since the models sometimes require much training before the effects of hyperparameter changes are observed, hyperparameter tuning has been limited. Selection of hyperparameter instead relied a lot on defaults in the tensor2tensor library as well as information from previous work on how to best train RNN-based seq2seq and Transformer models for summarization and translation. There are options for automatic hyperparameter tuning in the tensor2tensor library using the auto-tune flags. Auto-tuning does, however, require the models to be trained using Cloud ML Engine, which was not applicable as this would have incurred high costs.

Evaluating summary content is difficult as agreeing on what is essential in a text is subjective to some extent. By using the ROUGE metrics in conjunction with many human written reference summaries, this can be accounted for to some extent. Since the CNN/Dailymail data-set only contains one reference summary per article, this type of evaluation could not be made.

5.4 Future Work

Even though the training was conducted up until 500,000 steps in this thesis report, the change in ROUGE score after about 100,000 steps is limited for both of the models. One aspect that could be tested is how readability and grammar are affected as training progresses and how it correlates with the improvement in ROUGE score.

Another aspect that was not evaluated is how the readability of the generated summaries compares to the human-generated reference summaries in the CNN/Dailymail data-set. This would be interesting to explore as it could give further insight into the readability of the system generated summaries, however this evaluation was excluded as it would make the survey to extensive.

Since the Transformer is a relatively new model, it would be interesting to do a more extensive hyperparameter testing/tuning for summarization on either the Gigaword corpus or the CNN/Dailymail corpus. As this type of hyperparameter tuning has been done for machine translation [35] it would be possible to compare the results and eventually get new insights into the behaviour of the Transformer model.

5.5 Sustainability, ethics and societal aspects

Machine learning algorithms running in data centers around the world is starting to become a large part of the global energy consumption [40]. The total impact on sustainability is unclear since machine learning is also used in areas for predicting and managing energy consumption, helping to increase energy efficiency and reduce costs in many areas [41]. The models in this thesis report have been trained on Google TPUs which consumes less power than conventional GPUs [42].

When doing a survey, there are ethical aspects to consider, such as the collection and storage of personal information. A decision was made to minimize the amount of personal data collected as only data on education level was required by the participants.

As the models were trained on the non-anonymized version of the dataset, names and other information about individuals are still in the articles and will be displayed during the survey. Since this information has already been published in newspapers and the data set already being available online, it was considered a low risk that any sensitive data would be wrongfully distributed. There is a risk that the summaries produced by the models contain names and information in a context that is not coherent with the original article which could potentially be problematic. Another aspect, concerning the generation of automated summaries is that there is no guarantee that information is not distorted or that vital information is not omitted in the final summary. If automatically generated summaries were to be used to a more significant extent, this could have unknown societal effects.

Chapter 6

Conclusion

The Transformer has been evaluated against a seq2seq model with attention using both an automatic measure, ROUGE score, as well as using human evaluators.

For the first research question, "How does the Transformer model perform on the task of abstractive summarization in comparison with a seq2seq baseline, as measured with the ROUGE metric?", the results show that the Transformer achieves higher average ROUGE scores. For input lengths of 512 subwords the Transformer outperforms the seq2seq baseline by 9.41 ROUGE-1, 4.54 in ROUGE-2 and 8.85 in ROUGE-L when comparing average ROUGE scores. The Transformer also seems to be less affected by varying input length.

For the second research question, "How does the Transformer model compare with a seq2seq baseline when evaluating using human evaluators to measure readability and grammar of generated summaries?", the results show that the Transformer outperforms the seq2seq baseline in all of the categories evaluated. The largest difference being in the category of non-redundancy which is related to the issue of repetition when using the seq2seq baseline.

In addition to achieving higher evaluation scores, the Transformer trains faster and can train on longer input and output sequences before running out of memory.

Bibliography

[1] Mahak Gambhir and Vishal Gupta. “Recent automatic text summarization techniques: a survey”. eng. In: *Artificial Intelligence Review* 47.1 (2017), pp. 1–66. ISSN: 0269-2821. URL: `http://search.proquest.com/docview/1880005324/` (cit. on p. 1).

[2] Vipul Dalal and Latesh G. Malik. “A Survey of Extractive and Abstractive Text Summarization Techniques”. eng. In: IEEE, 2013, pp. 109–110. ISBN: 9781479925605 (cit. on p. 1).

[3] Abigail See, Peter J. Liu, and Christopher D. Manning. “Get To The Point: Summarization with Pointer-Generator Networks”. In: (2017) (cit. on pp. 1–3, 6, 7, 9, 12–14, 16, 17, 20, 29–31).

[4] Alexander M. Rush, Sumit Chopra, and Jason Weston. “A Neural Attention Model for Abstractive Sentence Summarization”. In: (2015) (cit. on pp. 2, 3, 5, 6, 12).

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural Machine Translation by Jointly Learning to Align and Translate”. In: (2014) (cit. on pp. 2, 7–9, 12, 30).

[6] Wenyuan Zeng et al. “Efficient Summarization with Read-Again and Copy Mechanism”. In: (2016) (cit. on pp. 2, 3, 6).

[7] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. “SummaRuNNer: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents”. In: (2016) (cit. on pp. 2, 4, 29).

[8] Romain Paulus, Caiming Xiong, and Richard Socher. “A Deep Reinforced Model for Abstractive Summarization”. In: (2017) (cit. on pp. 2, 6).

[9] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. “Sequence to sequence learning with neural networks”. In: *Advances in neural information processing systems*. 2014, pp. 3104–3112 (cit. on pp. 2, 7, 12).

[10] Ramesh Nallapati et al. “Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond”. In: (2016) (cit. on pp. 2, 3, 7, 9, 12, 14, 16, 20, 30, 31).

[11] Peter J Liu et al. “Generating wikipedia by summarizing long sequences”. In: *arXiv preprint arXiv:1801.10198* (2018) (cit. on pp. 2, 6, 13, 16, 18, 22, 29).

[12] Ashish Vaswani et al. “Attention Is All You Need”. In: (2017) (cit. on pp. 2, 10, 11, 13, 15, 18).

[13] Sumit Chopra, Michael Auli, and Alexander M Rush. “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*. 2016, pp. 93–98 (cit. on pp. 3, 14).

[14] Yau-Shian Wang and Hung-Yi Lee. “Learning to Encode Text as Human-Readable Summaries using Generative Adversarial Networks”. In: (2018) (cit. on p. 3).

[15] Chin-Yew Lin. “Rouge: A package for automatic evaluation of summaries”. In: *Text Summarization Branches Out* (2004) (cit. on pp. 3, 6).

[16] Asli Celikyilmaz et al. “Deep communicating agents for abstractive summarization”. In: *arXiv preprint arXiv:1803.10357* (2018) (cit. on p. 4).

[17] Qingyu Zhou et al. “Neural document summarization by jointly learning to score and select sentences”. In: *arXiv preprint arXiv:1807.02305* (2018) (cit. on p. 4).

[18] Kristian Woodsend and Mirella Lapata. “Learning to simplify sentences with quasi-synchronous grammar and integer programming”. In: *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics. 2011, pp. 409–420 (cit. on p. 5).

[19] Inderjeet Mani. “Summarization evaluation: An overview”. In: (2001) (cit. on pp. 6, 20).

[20] Chin-Yew Lin and Eduard Hovy. “Manual and automatic evaluation of summaries”. In: *Proceedings of the ACL-02 Workshop on Automatic Summarization-Volume 4*. Association for Computational Linguistics. 2002, pp. 45–51 (cit. on p. 6).

[21] Linqing Liu et al. “Generative Adversarial Network for Abstractive Text Summarization”. In: (2017) (cit. on p. 6).

[22] Michael Denkowski and Alon Lavie. “Meteor universal: Language specific translation evaluation for any target language”. In: *Proceedings of the ninth workshop on statistical machine translation*. 2014, pp. 376–380 (cit. on p. 7).

[23] Kyunghyun Cho et al. “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In: *arXiv preprint arXiv:1406.1078* (2014) (cit. on p. 7).

[24] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. “Effective approaches to attention-based neural machine translation”. In: *arXiv preprint arXiv:1508.04025* (2015) (cit. on p. 9).

[25] Yoshua Bengio et al. “A neural probabilistic language model”. In: *Journal of machine learning research* 3.Feb (2003), pp. 1137–1155 (cit. on p. 11).

[26] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018) (cit. on p. 13).

[27] Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. “Annotated gigaword”. In: *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*. Association for Computational Linguistics. 2012, pp. 95–100 (cit. on p. 14).

[28] Abi See. *cnn-dailymail*. `https://github.com/abisee/cnn-dailymail`. 2018 (cit. on p. 16).

[29] Rico Sennrich, Barry Haddow, and Alexandra Birch. “Neural machine translation of rare words with subword units”. In: *arXiv preprint arXiv:1508.07909* (2015) (cit. on p. 16).

[30] *Cloud Tensor Processing Units (TPUs)*. `https://cloud.google.com/tpu/docs/tpus`. Accessed: 2019-05-04 (cit. on p. 17).

[31] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on p. 18).

[32] Noam Shazeer and Mitchell Stern. “Adafactor: Adaptive learning rates with sublinear memory cost”. In: *arXiv preprint arXiv:1804.04235* (2018) (cit. on pp. 18, 30).

[33] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958 (cit. on p. 19).

[34] Priya Goyal et al. “Accurate, large minibatch sgd: Training imagenet in 1 hour”. In: *arXiv preprint arXiv:1706.02677* (2017) (cit. on p. 19).

[35] Martin Popel and Ondřej Bojar. “Training tips for the transformer model”. In: *The Prague Bulletin of Mathematical Linguistics* 110.1 (2018), pp. 43–70 (cit. on pp. 19, 21, 32).

[36] *Cloud TPU System Architectu.* `https://cloud.google.com/tpu/docs/system-architecture`. Accessed: 2019-05-04 (cit. on p. 20).

[37] mikeymezher. *TPU failing to reinitialize after eval.* `https://github.com/tensorflow/tensor2tensor/issues/1202`. Accessed: 2019-05-04. 2018 (cit. on p. 20).

[38] *py-rouge 1.1.* `https://pypi.org/project/py-rouge/`. Accessed: 2019-05-04. 2018 (cit. on p. 20).

[39] Hoa Trang Dang. “Overview of DUC 2005”. In: Citeseer (cit. on p. 22).

[40] Eva Garcia Martin. “Energy Efficiency in Machine Learning : A position paper”. In: *30th Annual Workshop of the Swedish Artificial Intelligence Society SAIS 2017, May 15–16, 2017, Karlskrona, Sweden :* vol. 137. Linköping Electronic Conference Proceedings 137. 2017, pp. 68–72. ISBN: 978-91-7685-496-9 (cit. on p. 33).

[41] Josep Ll Berral et al. “Towards energy-aware scheduling in data centers using machine learning”. In: *Proceedings of the 1st International Conference on energy-Efficient Computing and Networking*. ACM. 2010, pp. 215–224 (cit. on p. 33).

[42] *What makes TPUs fine-tuned for deep learning?* `https://cloud.google.com/blog/products/ai-machine-learning/what-makes-tpus-fine-tuned-for-deep-learning`. Accessed: 2019-05-28 (cit. on p. 33).

Appendix A

Appendix

A.1 Summary Examples

In table A.1, A.2, A.3 some examples are shown displaying the summary generation from the models. The example summaries are generated from the test set on a max length of 256.

Table A.1: Example generated text 1

Source	Real Sociedad remain hopeful over a deal for Burnley striker Danny Ings. Liverpool, Manchester City and Manchester United have all shown interest in the England Under 21 international but have yet to sign him up as they consider other options also. Ings has interest from Borussia Monchengladbach among others but Sociedad have remained constant from the start. . Real Sociedad are still hopeful of singing Burnley striker Danny Ings when his contract expires in the summer. Michel Vorm saves from Ings as the Burnley forward passes up a great chance to give his side the lead against Tottenham on Sunday. England Under 21 international Ings chats with Harry Kane after the final whistle at Turf Moor. The 23-year-old knows he can play regularly there, will be watched with interest by England coach Roy Hodgson and will get the money he wants. Sociedad manager David Moyes has taken the club into the top half of La Liga and intends to build on that in the summer and Ings could prove a key acquisition. Ings' contract at Burnley expires in the summer and is unlikely to sign another one regardless of their ability to survive relegation this season. Should Ings move to another Premier League club Burnley would be due a sizeable compensation fee because of his age, but that fee would be nominal if the striker moved abroad. .
Gold Standard	Danny Ings' contract expires in the summer and is set to leave Burnley. Real Sociedad want to take Ings to La Liga on a free transfer. Manager David Moyes faces competition from Liverpool among others.
LSTM	Real Sociedad remain hopeful of signing Danny Ings in the summer. Ings has interest from Borussia Monchengladbach and Sociedad. The 23-year-old has interest from Borussia Monchengladbach and Sociedad. Click here for more Real Sociedad transfer news .
Transformer	Real Sociedad are still hopeful of singing Danny Ings when his contract expires in the summer. Liverpool, Manchester City and Manchester United have all shown interest. Ings' contract at Burnley expires in the summer and is unlikely to sign another Premier League club. READ: Real Sociedad are still hopeful of singing Burnley striker Danny Ings. Click here for more transfer news .

Table A.2: Example 2

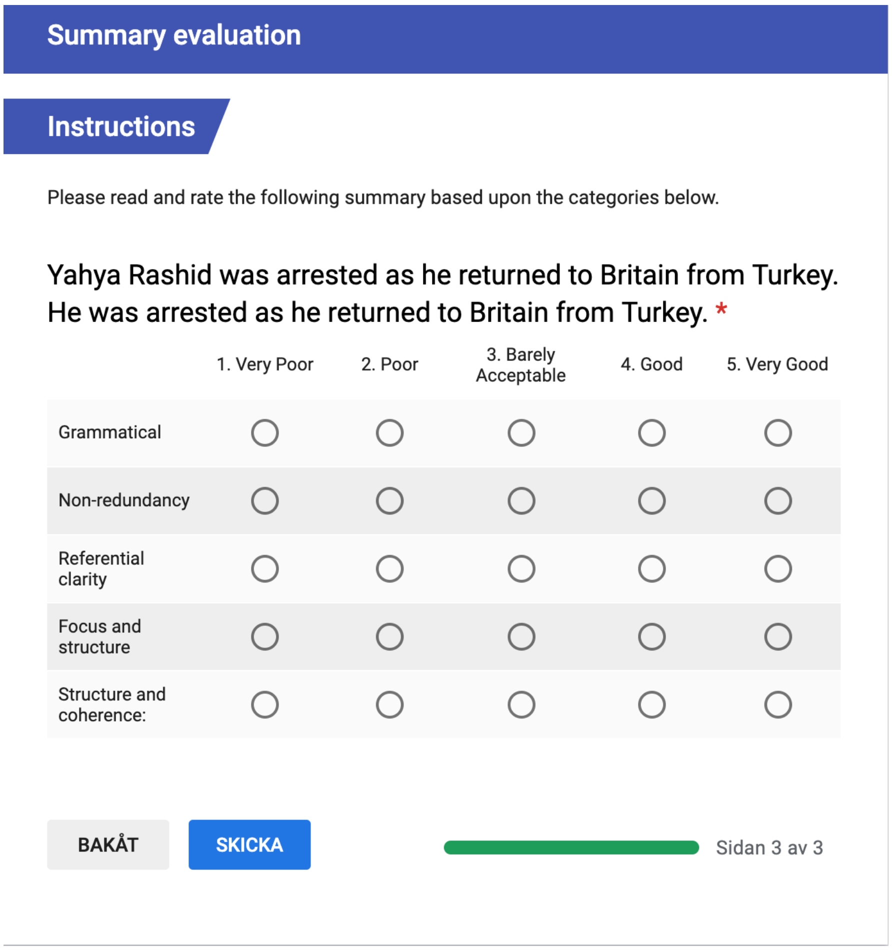
Source	Welsh international Elliot Kear is back in rugby league after signing for Championship club London Broncos. The 26-year-old winger or full-back has spent most of the year playing rugby union for London Welsh after leaving Bradford at the end of last season. 'I'm really pleased to sign the contract and come back to rugby league,' said Kear. 'I'm just really excited to get back on the field. Elliot Kear will return to rugby league's Championship with London Broncos after a season with Welsh. The Wales rugby league international switches back to the code after London Welsh were relegated. 'I know a few of the boys from playing against them over the years and everyone has been really welcoming.' London coach Andrew Henderson said: 'It's a great signing for the club. What he will bring is a vast amount of experience at Super League level and Championship level from when he was first starting out. 'He knows what this level of football is about as well. He's a Welsh international too so he's played at the highest level and some of the biggest stages in the world.' Kear will be available for Sunday's home game against Dewsbury. Kear moved to Premiership Rugby from Bradford and returns for the Broncos under Andrew Henderson.
Gold Standard	Elliot Kear played for Premiership Rugby's relegated London Welsh. He joined the Oxford-based rugby club from Super League's Bradford. Kear could line up for the Broncos as soon as Sunday against Dewsbury.
LSTM	Elliot Kear has spent most of the year playing rugby union for London Welsh after leaving Bradford at the end of last season. The 26-year-old winger has spent most of the year playing rugby union for London Welsh after leaving Bradford at the end of last season. He will be available for Sunday's home game against Dewsbury.
Transformer	Elliot Kear will return to rugby league with London Broncos. The 26-year-old winger has spent most of the year playing rugby union. Kear moved to Premiership from Bradford and returns for the Broncos.

Table A.3: Example 3

Source	Bayern Munich were in high spirits after they restored their lead at the top of the Bundesliga back to 10 points. Pep Guardiola put his players through their paces as they carried out Rondo passing drills in training. Xabi Alonso was left stretching for a pass with his team-mates watching on and ending up falling over, as the ball approached him at pace. VIDEO: Watch Xabi Alonso fall over during 'Rondo' training drill . Xabi Alonso fails to control a pass during the Rondo drill and loses his balance. The Bayern Munich midfielder takes a tumble and the ball flies off in the opposite direction. The former Liverpool midfielder has his hands on his heads on the ground after the incident. Alonso is flat on his back as his team-mates watch on and see the funny side to his fall. The one touch football routine has become a fundamental part of life under Guardiola and the players have clearly warmed to his methods. Robert Lewandowski returned to haunt at the Signal Iduna Park on Saturday as Bayern beat Borussia Dortmund. Bayern suffered a shock 2-0 home defeat by Borussia Monchengladbach before the international break but responded to get their title charge back on track against their former championship rivals. . .
Gold Standard	Xabi Alonso has been in fine form for Bayern Munich. Robert Lewandowski ensured Bayern beat Dortmund on Saturday. Pep Guardiola's side are top of the Bundesliga by 10 points.
LSTM	Bayern Munich beat Borussia Dortmund 2-0 at the Signal Iduna Park. Xabi Alonso fails to control a pass during the Rondo drill. Bayern beat Borussia Monchengladbach 2-0 at the Signal Iduna Park. Bayern beat Borussia Monchengladbach 2-0 at the Signal Iduna Park. Bayern beat Borussia Monchengladbach 2-0 at the Signal Iduna Park. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 at Signal Iduna Park on Saturday. Bayern beat Borussia Monchengladbach 2-0 before the international break.
Transformer	Bayern Munich lost 2-0 to Borussia Monchengladbach on Saturday. Xabi Alonso was left stretching for a pass with his team-mates. Pep Guardiola put his players through their paces as they carried out passing drills. The former Liverpool midfielder is flat on his back as his team-mates watch on and see the funny side to fall. Robert Lewandowski returned to haunt at Iduna Park on Saturday.

A.2 Survey design

Figure A.1 shows the design of the survey questions.



Summary evaluation

Instructions

Please read and rate the following summary based upon the categories below.

Yahya Rashid was arrested as he returned to Britain from Turkey. He was arrested as he returned to Britain from Turkey. *

	1. Very Poor	2. Poor	3. Barely Acceptable	4. Good	5. Very Good
Grammatical	○	○	○	○	○
Non-redundancy	○	○	○	○	○
Referential clarity	○	○	○	○	○
Focus and structure	○	○	○	○	○
Structure and coherence:	○	○	○	○	○

Figure A.1: Evaluation form excerpt

A.3 Survey results

Figure A.2 shows the distribution of points for each category in the survey.

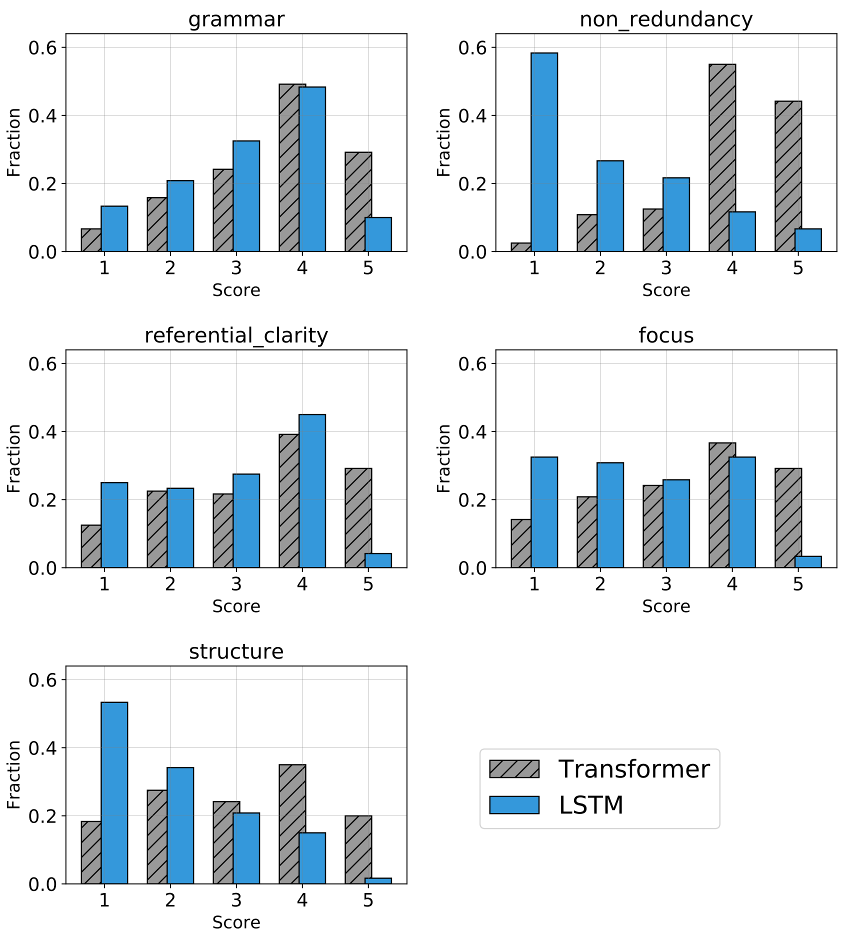


Figure A.2: Showing survey results for each category for both the Transformer and the LSTM model. The y-axis shows the fraction of points recieved for that category and the x-axis shows the points from 1-5.

TRITA -EECS-EX-2019:563