Master Thesis

Multi-lingual Transfer in Automatic Summarization

Omendra Kumar Manhar

Master Thesis DKE-20-07

Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science of Artificial Intelligence
at the Department of Data Science and Knowledge Engineering
of the Maastricht University

Thesis Committee:

Dr. Jan Niehues
Dr. Jerry Spanakis
Dr. Laura Astola (Accenture B.V. (external))

Maastricht University
Faculty of Science and Engineering
Department of Data Science and Knowledge Engineering

January 26, 2020

Acknowledgements

This thesis was done in partial fulfilment of Masters program in Artificial Intelligence. This was an external internship conducted with Accenture, Heerlen. I would like to thank my supervisors Dr. Jan Niehues and Dr. Laura Astola (Accenture) for their constant guidance and allowing me to work and learn in a productive environment.

I would like to add that I was provided with resources and constant feedback from the aforementioned supervisors.

Accenture, Heerlen has a team which works in Data Science related projects, they always had positive and constructive feedback for me. I was given a lot of freedom in terms of selection of the thesis. Accenture also provided a very friendly and approachable work environment for me to explore solutions and estimate their feasibility under different environments.

Abstract

Automatic summarization is one of the most researched fields in Natural Language Processing domain. State-of-the-art summarization[20][24] models are capable of generating readable and meaningful summaries to match how a human would summarize a text. However, these models have been trained on English Dataset to generate English summaries. Currently with the rise of availability of data in multiple languages, there is room for the automatic summarization models to expands from singular language to generate readable meaningful summaries in multiple languages simultaneously.

This research investigates various approaches to achieve multilingual automatic summarization models, while having only an English-to-English summarization training set. First, a cascaded approach is employed to serially process a text using multiple summarization and translation models. Each of these models architecture is based on transformer models[33]. *Webhose* dataset has been used to evaluate the performances of the summarization approaches in this research. It should be noted that this research is the first scientific work to evaluate summarization models on this datatset. Cascaded approach were evaluated on *webhose* corpus with an average ROUGE(R1) score of 9.70. Second, a multitask learning model is designed to train a machine translation model to perform summarization and translation at the same time. In order learn the representation of both tasks, input and output sequences are tagged with symbols to represent summarization, translation and languages. However, the results from multitask models generated summaries containing phrases from multiple languages, hence evaluating considerably lower at ROUGE(R1) score of 2.54. The multitask model suffers from a bias towards the training set due to only having summarization training set containing English to English summaries.

Contents

Chapter 1

Introduction

Recent developments in the Natural Language Processing and Machine Translation have given rise to many venues of exploration in terms of language understanding and modeling. However, up until now majority of the text generation models are based on English language. This is mainly due to the readily available free datasets across various providers, it is much easier to find datasets in English compared to other languages. However, through various web scraping tools, it is possible to find text datasets in other languages.

With the exponential rise of data availability in all sorts of domains in different languages, the tools and methods available in big data and summarization are proving to be useful in order to get insight into huge chunks of text without actually having to go through all of it manually. This research focuses on using these models to evaluate whether it is possible to achieve comparatively good results (in automatic summarization) when the training dataset is not in English. It should be noted that the dataset for summarization is only in English, however in order to make the model learn the translation task in as well, a dataset containing parallel corpus for different languages (specifically in German and English) has been has also been used. One of the main objectives of this research is also to implement machine learning models to learn the basic representation of translation and summarization related tasks independently and also simultaneously.

Following research question were formed during the initial phase of planning:

- Is it possible to build a cascaded system of various translation and summarization models to work serially and achieve multilingual summarization?
- Can a machine translation model be trained to perform summarization and translation in a multi-task learning method?
- Is it possible for multitask learning model to summarize non English texts to non English summaries?
- How much knowledge is transferred and lost during the simultaneous application of these transduction models on text?

The recent popularity of modern text transduction models[33] has presented a multitude of opportunities to quantitatively answer such questions. The performance of text transduction models are evident when it comes to translation or summary generation (strictly in English), however there hasn't been a lot of attention towards generating summaries in different languages. Given that there exists a whole chunk of knowledge resources in multiple languages, it is a very important issue that should be explored further in detail using the state-of-the-art text transduction model. Also to discover whether there can be just an efficient of summarization when the input text is not in English(original language of the training source).

Chapter 2

Basics/ Related work

2.1 Automatic Summarization

Automatic summarization[16] is the process of summarizing text using natural language processing tools into smaller chunks of text while preserving the initial information and meaning that was conveyed in the original content. For any text summarization model, the end goal is to generate summaries as if they were written by a human. The closer it is to a manual summary the higher it will be evaluated as an auto generated summary. In automatic summarization there are various ways of generating summaries for a given text, generally there are two approaches of designing automatic summarization models.

2.1.1 Extraction based summarization

In this type of summarization the summary is generated by picking sentences, phrases or words from the original text and combining them together. In this method the original sentences are not modified in any way, rather they are selected based on their importance to the entire text. The main objective in extraction based summaries is selecting content that should remain in the final output summary. This elimination of redundant information and selection of most important content is done by weighing different parts of the sentences against other, and there are many techniques available to perform those computations on the sentences[9]. This approach is about learning the importance of the respective sentences and how they relate to each other.

The primary steps in extraction based summarization can be defined as below [6]:

- Apply a text cleaning technique if desired before splitting the original text into sentences.
- After extracting sentences, they can be processed to remove stop words (stop words in Natural Language Processing refers to the set of words which occur too often in a sentence)

- Tokenize the sentences received from above step, to finally have a collection of words from the text
- Calculate the weighing frequency of each words in the list from above [21]
- From the word weights above, weights of each sentence can be calculated by simply adding the weights of the words. Sentences with higher weights will be selected to stay in the summary

2.1.2 Abstraction based summarization

Unlike extraction based summarization, this method is about understanding the content of the text and then creating the summary using new sentences. This method generates it's own sentences to represent the content of the original text much like how a human would summarize the text. Since the sentences are generated to condense the information provided in the original text by somewhat paraphrasing the sentences, this complexity makes this approach much more challenging than the aforementioned Extraction based approach.

In this research, the summarization models built are based on abstraction based approach. There have been multitude of scientific works in terms of summarizing texts in a singular language, e.g. generating an English summary from English text using state of the art text generation models[5]. However this research examines various methods to generate summaries in multiple languages, specifically in English, German and Dutch from an original text which could be in either of these three aforementioned languages. It should be noted that the summarization training dataset is strictly in English. It is important to evaluate the knowledge transfer of the learned summarization and translation methods.

2.2 Text processing

As in any machine translation and natural language processing method, preprocessing is an imperative aspect to how the training model behaves and performs upon evaluation phase. In traditional word embedding methods such as *Word2Vec*[17] models, the texts are processed in word level and they are modelled using skip-gram model or continuous bag-of-words model. Each word has a unique vector representation mapped to itself. Since, these models are dependent on words and their availability in the training set, they disregard the internal structures of the words and their formations, for example, the relationship between the words "run", "runs" and "running". So, when the embedding model encounters an unseen word which was not present in the original training set, it usually throws an exception or substitutes it with a default token for an unknown word. Additionally, storing every word as a unique token is an expensive process, especially if the training dataset consists of corpus in multiple languages.

In order to employ an efficient tokenization method to handle unseen words, a subword model[13] is used. Subword model tokenize the sequences by splitting words into unique subwords. One such method is called Byte pair encoding (BPE) [28]. Byte Pair Encoding is a form of data compression in which the most common pair of consecutive bytes of data is replaced with a byte that does not occur within that data [28]. This way of text representation steers the dependence from word level for tokenization of sentences to unique tokens present in the dataset. BPE model construction generally done by following steps:

- Prepare a text corpus to train the BPE model on
- Define a vocabulary size (N_{voacb_size})for the model to evaluate the number of subwords
- Split all words into characters while noting the frequency of the word they belonged to
- Create a new subword by considering high frequency sequence of characters from the previous step
- Repeat this process until the number of subwords generated reaches N_{vocab_size}

Tokenizer model such as BPE learns the internal structures of the words. It is language independent model by design since it depends on subword units. It learns based on the unique tokens and bytes present within the training set. The vocabulary size is already defined before the training of the model starts, the vocabulary size determines the token sizes when encoding the sentences as tokens, meaning that a higher vocabulary size allows for more individual tokens to be stored in the vocabulary based on their individual frequencies.

2.3 Recurrent Neural Networks (RNNs)

Recurrent Neural Networks[18] are neural networks which are modeled to utilize sequential information. These networks are designed to produce outputs sequentially so that the output is dependent on the previous computations made on the sequence, it is an improvement over a typical neural network which considers all inputs to be independent of each other. This property enforces a dependency over the past computations which can be considered as a memory for the network, or an ability to look back in the past steps.

However RNNs can only look a few steps in the past. They suffer from vanishing gradient problem and they fail to perform accurate translation due to lack of available memory needed to be able to accurately perform neural machine translation tasks for longer sentences [10]. In order to generate automatic summaries from long texts, it becomes important that the machine translation and summarization architectures employs a memory technique to consider the entire sequence while generating outputs .Their long term dependency problem is somewhat addressed by a modified form of recurrent neural network called Long Short Term Memory models (LSTMs)[11].

However, despite its minor improvement over traditional RNNs, LSTMs still fail to keep the context for a word that is too far away from the current word, because the probability of keeping it decreases exponentially with distance. Additionally due to their sequential nature do not allow for parallelization to speed up training processes which presents an obstacle for training over larger datasets.

Transformer models were designed to overcome the problems faced when using Recurrent Neural Networks (RNNs), Transformer models address them by using feed forward networks with attention models. It does not need to process sentences in order (beginning to end), which allows for more parallelization to speed up the training process and modeling, meaning faster translation. The attention mechanisms are there to mathematically represent the importance of tokens, in order to preserve the context when generating output sequences. The principle behind this is that every word could provide meaning and context in a sentence. In this research these models are leveraged to build translation and summarization models together and combine them to perform evaluations on a multi-lingual dataset.

2.4 Encoder-Decoder architecture

Encoder - Decoder architecture[3] in machine translation containing separate encoder and decoder stacks consisting of recurrent neural networks. Figure 2.1 represents an overview of an encoder-decoder architecture. This model is developed to first process an input entirely or serially at the encoder stack and encode it into some form of an internal representation of the input. Once encoding of the input sequences is done, the decoder networks uses it to generate texts using the learned representation through learning over a training set. *Sequence2Sequence*[31] models use recurrent neural networks as encoders and decoders to perform various machine translation tasks. The encoding an decoding mechanism can be modified to further enhance the performance of the model, for example replacing RNNs with convolutional neural networks [7].

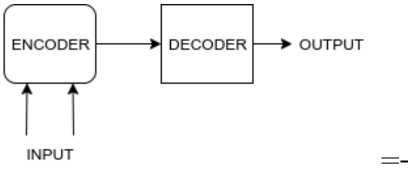


Figure 2.1: A simple encoder - decoder architecture

2.5 Attention mechanism

One of the issues with the encoder-decoder architecture briefly described in section 2.4 is that for every input sequence, no matter what the size it works by compressing them all into a vector of fixed size. This can be potentially problematic when the size of the neural network and the size of the input sequences are not suitable (smaller neural network for longer input sequences), information could be lost while processing longer sentences.

Attention models[2] addresses this issue by letting the model learn a vector representation for each input word. So, the model learns representations efficiently by moving away from fixed length vector representation. This leads the model to learn which part of the sentence should be attended and allows the decoder to attend to different parts of the input sequence. An attention matrix consisting of mapping of input and output words is created by using the learned representation/context vectors.

Figure 2.2 shows the attention mechanism model. Here encoder generates $h_1, h_2, h_3..h_n$ from input sequence $x_1, x_2, x_3, ..x_T$. Here a is an alignment model which is essentially a feed forwards network. Alignment model scores represent the encoded input$(h_1, h_2, h_3..h_T)$ relation with the decoder's output(s) [2].

The representation vector is calculated as the weighted sum of the encoded input sequence $(h_1, h_2, h_3..h_n)$ (refer equation 2.1).

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_i \tag{2.1}$$

where

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \tag{2.2}$$

and the alignment scores e_{ij} are applied with a softmax function for normalization purposes.

$$e_{ij} = a(s_{i-1}, h_j) \tag{2.3}$$

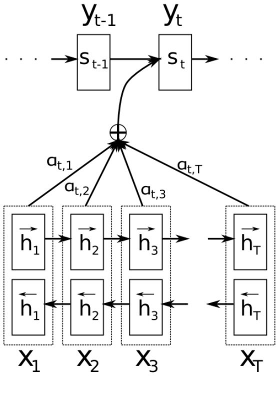


Figure 2.2: Attention mechanism ([Bahdanau et al., 2014])

2.6 Transformers

Recently in the field of machine translation and text reduction a neural network architecture called Transformer models are gaining popularity due to their significant advantages over the sequential neural network architectures. Transformer models are attention based models which are widely popular in Natural Language Processing field for machine translation and text generation tasks. They are designed similarly to sequence-to-sequence models, as in they are provided with a sequence and are trained to generate another sequence which could be either a translated text or generated summary. The transformer models shows that it is possible to move away from using recurrent neural networks within the encoder and decoder stacks by using attention mechanisms instead. This makes room for parallelization during translation and training processes, making these models to be considerably more efficient and capable than the RNNs / LSTMs [30].

Like most sequence transduction models, transformers also follow suit with having an encoder-decode type architecture. Figure 2.4 represents the transformer architecture . The transformer models consist of $N = 6$ stacked identical layers of encoders and decoders [33]. The input text sequences to the encoders are turned into vector representations by using an embedding algorithm. However, since transformer models do not process inputs sequentially, there needs to be a way to contextualize the order of the words in sequences. This is solved by adding Positional Encoding to the input embeddings at the bottom of encoder and decoders each, so it is imperative for positional encoding to have the same size as the input embeddings, so they can be added before to being supplied to the encoder/decoder stack.

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}}) \tag{2.4}$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}}) \tag{2.5}$$

Equation 2.4 and 2.5 show the constant created to represent the position specific values. Here *pos* refers to the position and i refers to the position along the embedding vector dimension.

$$Input = Embedding_{(seq,d_{model})} + PositionalEncoding_{(seq,d_{model})} \tag{2.6}$$

Equation 2.6 shows how the input at the bottom of the encoding component looks like before it is passed on to the first encoder layer. Each encoder layer consists of two sub-layers called self attention layer and position wise feed-forward layer.

Attention layers are designed to encode the words based on all the other words in a particular sequence. Figure 2.3 demonstrates the structure of the attention layer. K, Q and V are Key, Query and Value vectors respectively. They are calculated by packing the input embeddings into a matrix and multiplying them by the weight matrices that have been trained so far, e.g. $Q = X * W_Q$. One the Key, Value and Query Vectors have been calculated, equation 2.8 can be used to calculate attention. This is also termed as ”scaled dot product attention. The input contains keys and queries each of dimension d_k and values of dimension d_v. Afterwards a softmax function is applied to the values. Another approach to compute attention apart from scaled dot product attention, is additive attention. In this mechanism, the attention is calculated through a feed forward layer with a hidden layer [2]. However ”scaled dot product attention” is more efficient and faster in practice, since it can be implemented using optimized matrix multiplication techniques. Using multi-head attention it is possible to compute different sets of query, values and keys for the same original embedding. Attention is computed on each of these sets separately and concatenated together and multiplied with another learned matrix Z to reduce these embedding to a single embedding (Equation 2.7). Each of these new embedding sets represent a head of the self attention model, which is why this is called multi-head attention [33].

$$MultiHead(Q, K, V) = concatenate(h_1, h_2, ...) * Z \tag{2.7}$$

$$Attention(Q, K, V) = softmax(QK^T/\sqrt{d_k})V \tag{2.8}$$

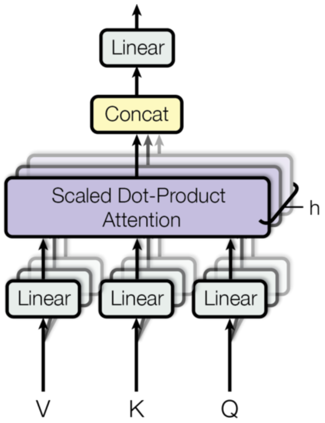


Figure 2.3: Self Attention Layer ([Vaswani et al., 2017])

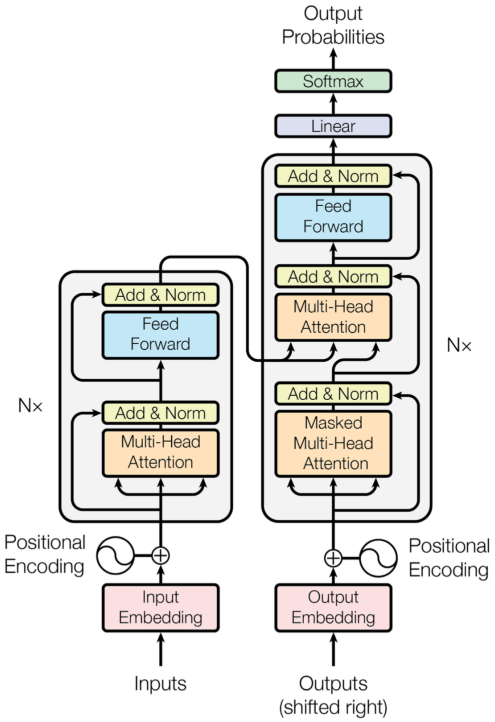


Figure 2.4: Transformer Model Architecture ([Vaswani et al., 2017])

Additionally, Apart from self-attention model, each of the encoder and decoder layers consist of a fully connected feed forwards neural network which contains two linear layers with a ReLU activation between them. It is applied in each position independently and identically [33].

$$FFN(x) = max(0, x * W_1 + b_1) * W_2 + b_2 \tag{2.9}$$

Here in equation 2.9, x represents a set of embeddings. As consistent throughout the transformer architecture the dimensionality of input and output is the same $d_{model}(= 512)$ while the inner layer dimensionality is $d_{ffn}(= 2048)$ [33].

Layer Normalization[1] is applied to normalize the input, across each of the two sublayers. In this model it is done by residuals (see equation 2.10).

$$SubLayer(x) = LayerNormalization(x + SubLayer(x)) \tag{2.10}$$

Decoder layers in transformers are identical to encoders but with some modification to the self attention layer. Additionally, decoders also have a third layer which executes multi-head attention over the overall output from the encoder. The self attention layers in decoders are modified to prevent the decoder from having access to words that come after the word its attempting to predict at a position. This is done by masking the embeddings for all those words by multiplying them by 0, making sure that the prediction only depends on the words that already came before it. Similar to the encoder stack mechanism, each sublayers in decoder stack are also followed by layer normalization [33].

Finally, from the decoder stack a vector is produced, which are should now be converted back to words to generate sentences. This is done by employing a linear and a softmax layer at the top of the decoder stack. This linear layer is a fully connected neural networks that takes the output vectors and projects them into a larger vector called logit vectors. These logit vectors have the dimensionality of the vocabulary size from the initial training set, hence each cell indicating score for that unique token. A softmax function is applied at the end to convert all of these scores into probabilities and the ones with the highest are chosen and mapped with the corresponding tokens for each time step to eventually form a sentence over the time steps. The final result can be decoded into words using the subword algorithm (byte pair encoding) that was applied during the input training set preparation.

2.7 State-of-the-art summarization model

[Nallapati et al., 2016] used encoder-decoder architecture with recurrent neural networks employed with attention mechanism to achieve state-of-the-art text summarization results on two different datasets (*CNN/Dailymail*[4] and *Gigaword*). They proposed various models within the encoder-decoder architecture to address specific issues within the abstractive summarization models such as improving the efficiency by reducing the size of softmax layer in the decoder and hence making it more feasible computationally speaking. They also modified the encoder to be able to capture keywords in a document which is the main theme of the document. The proposed models were shown to bring improvement over the traditional architectures[20].

[See et al., 2017] presented a model to overcome the issues faced when summarizing text through sequence to sequence models. The shortcomings were that sometimes the factual information from the original texts were not being accurately replicated in the resulting summaries, and they are prone to repeating themselves. Their model consists of hybrid-pointer generator network that points to certain words to be replicated in the resulting summary, thereby retaining the factual information. While the generator network makes sure that novel words are generated during the summary generation. In order to prevent repetition they kept track of the what has been already summarized by employing a coverage mechanism. This is to lower the probability of already generated content to be repeated [27].

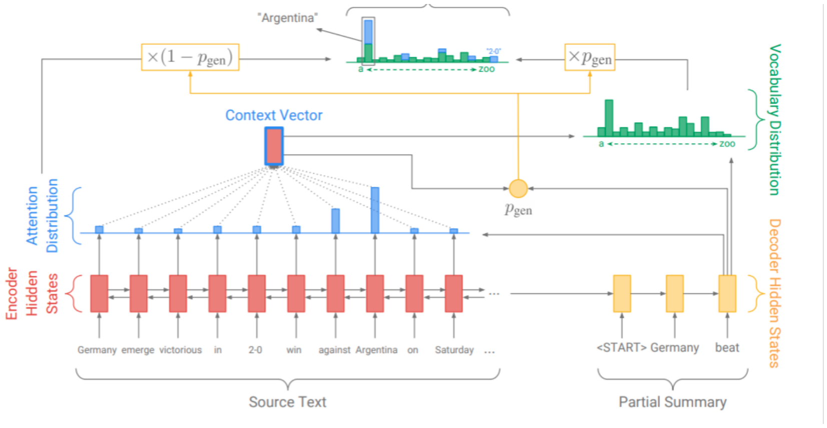


Figure 2.5: Pointer-Generator Network ([See et al., 2017])

The model was applied to the CNN/DailyMail dataset (see chapter 3), and was able to achieve ROUGE(R1) score of 39.53.

[Paulus et al., 2017] presented an intra-attention neural network model that was designed to attend over the input and output sequences individually, and employed a training mechanism to utilize traditional word prediction learning with reinforcement learning. Their main goal was to produce a more readable summary than the traditional word prediction learning method does (they tend to exhibit bias towards ground truth). Their model consists of an encoder based on a bi-directional LSTM while using an LSTM as a decoder. The encoder's job is to calculate hidden state h^e from input embedding x_i, while the decoder computes hidden states h^d from an output embedding y_i, the embeddings are taken from an embedding matrix $W_{embedding}$ [24]. They tested their model on two datasets (*New York Times* and *CNN/DailyMail*), they were able to achieve a ROUGE(R1)score of 41.16.

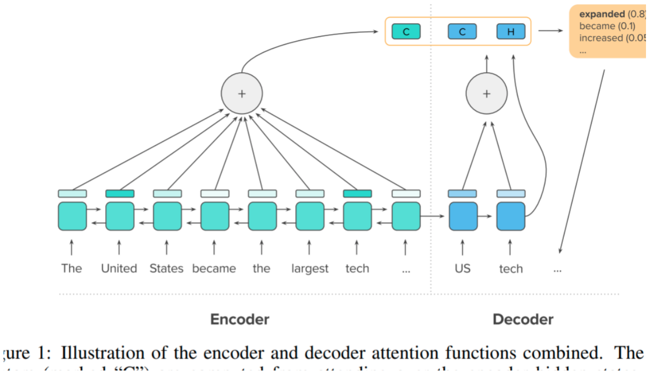


Figure 2.6: Encoder-Decoder with attention ([Paulus et al., 2017])

[Gehrmann et al., 2018] proposed a model to address the issue with most summarization models which are regarding selecting the important key-points within the original text. They showed an improvement in evaluation score by designing a content selector to choose which information of the source must be part of the summary as well, hence restricting the summary content in terms of sentences[8].

Unlike the other works mentioned above, [Liu et al., 2018] approached abstraction based summarization using a Generative Adversarial Network model[25]. They employed a generator model which generates an abstraction based summary from an original text which is evaluated and rated by a discriminator model based on how close it is from the expected output. Over time this model was able to achieve competitive results over the *CNN/DailyMail* dataset[15]. They were able to achieve ROUGE(R1) scores of 39.91.

Chapter 3

Dataset

Obtaining an appropriate dataset for training a machine learning model is one of the most important parts of the model building process. The performance of the model hugely depends on the size, type and the variance in the dataset. One of the first objectives for this research were to acquire freely available datasets containing original texts and their manual summaries. The training and testing dataset mustn't be auto generated text but manual texts from real-life resources such as news websites, reviews, etc. This particular requirement suits perfectly for databases that store news articles and their titles. There were three sets of databases used for the experiments and evaluations performed within the scope of this research.

Dataset	Languages	Format	Articles
News Commentary	English, German, Dutch	.tsv	57906
CNN/DailyMail	English	.txt	311971
Webhose.io	English, Dutch, German	.json	575330

Table 3.1: Dataset available

3.1 CNN/DailyMail

CNN/DailyMail[27][4][32] dataset is one of the popular resources available for training text summarization models. It consists of news articles with almost each article content accompanied with tagged headlines. These headlines are put together and serve as a consolidated summary of the corresponding original news article. There are various ways to extract this dataset, however this dataset was obtained directly by extracting the dataset using the resource links that was provided in the work done by [See et al., 2017]. dataset, extracted using a simple extraction tool, it contains text files which are named accordingly for the use. Each line contains one article, with the text content under the tag *summ-content* and the corresponding summary is under *s* tags. A simple reading of the file line by line while separating the tags can easily create a training set.

This entire dataset is in a singular language (English to English), however it serves as a great training set for the examining the multi-lingual transfer in summarization models, once the models have learned the summarization task separately. The extracted dataset is split into 287226 training examples, 13,368 validation examples and 11490 test examples. There are on average of 781 words per article with 56 words per summary.

3.2 Webhose

Webhose.io[34] is one of the providers of web content such as blogs, reviews and news articles in a readable data format such as ”json”, specifically catered towards data science and data mining techniques. It provides a multitude of free to use text datasets in different languages (English, Russian, Dutch, German), which are generally scraped from news articles. They can be used as a testing set for text transduction model. It does not contain parallel corpus, which makes it harder for it to serve as training set for a translation model. However they can be used to evaluate summaries, as every article contains a title and also a description. These articles in different languages are used as a testing dataset for the models that have been implemented for this research. The articles provided are an individual ”json” file per article, with clearly defined labels and values. Using the downloaded compressed file, all the json files are extracted and loaded using python data loaders. For each article, the only information that was of pertinence to this research were ”text”(body of the article), ”headline” and ”highlights”. Rest of the information such as ”authors”, ”date” and url can be discarded. This was done by individually processing all the ”json” files (see table 3.2 for more details) and sorting the ”headlines”, ”highlights” and ”text” out to be dumped into a consolidated ”testing” dataset.

Language	Examples	Source Tokens(Average)	Target Tokens(Average)
German	398840	292	9
Dutch	116193	176	8
English	60297	429	10

Table 3.2: Webhose Dataset details

This dataset is not used for training either of the translation or summarization model, it is used for testing the automatic summaries generated by the different approaches employed. Comes with independent news articles in German, Dutch and English with titles and summaries, making it a novel dataset to achieve machine translation and summarization performance evaluation. The details regarding each of the corpus for English, German, and Dutch can be found in table 3.2.

```
{
   "title": "Frank Witzel erhält Deutschen Buchpreis 2015 - Rhein-Neckar-Zeitung Regionalnachrichten - Rhein Neckar Zeitung",
   "text": "Artikel Frank Witzel erhält Deutschen Buchpreis 2015 Frankfurt/Main (dpa) - Für einen Roman über die alte
        Bundesrepublik hat Frank Witzel den Deutschen Buchpreis 2015 erhalten. Das Buch mit dem Titel Die
        Erfindung der Roten Armee Fraktion durch einen manisch-depressiven Teenager im Sommer 1969 wurde in Frankfurt
        als beste literarische Neuerscheinung des Jahres im deutschsprachigen Raum ausgezeichnet. Der Autor schildert darin in
        einer Vielzahl von Episoden und Fragmenten die Nachkriegszeit aus der Sicht eines 13-Jährigen im Wiesbadener
        Ortsteil Biebrich. Der Deutsche Buchpreis, 2005 erstmals vergeben, gilt als wichtigste Auszeichnung der Branche. 12.10.2015,
        19:16 Uhr Frankfurt/Main (dpa) - Für einen Roman über die alte Bundesrepublik hat Frank Witzel
        den Deutschen Buchpreis 2015 erhalten. Das Buch mit dem Titel Die Erfindung der Roten Armee
        Fraktion durch einen manisch-depressiven Teenager im Sommer 1969 wurde in Frankfurt als beste literarische Neuerscheinung
        des Jahres im deutschsprachigen Raum ausgezeichnet. Der Autor schildert darin in einer Vielzahl von Episoden
        und Fragmenten die Nachkriegszeit aus der Sicht eines 13-Jährigen im Wiesbadener Ortsteil Biebrich. Der Deutsche
        Buchpreis, 2005 erstmals vergeben, gilt als wichtigste Auszeichnung der Branche. weitere Meldungen"
}
```

Figure 3.1: Webhose Data Sample

Figure 3.1 shows a part of the dataset for German language, in each of the json file the *title* and *text* are pertinent keys to this research. This format is consistent across all corpus in different languages. One of the notable things about this dataset for all three languages, is that they do not have *highlightText* and *highlightTitle*, making it essentially a title and text based corpus. It is also to be noted that there are articles which describe either a metadata such as date, newspaper name or just a phrase to describe the nature of the article (for example: breaking news). It was not possible to manually check how many of those json files exhibit such titles, however figure 3.2 demonstrates one such example.

```
{
   "title": "+++ BILD Breaking News +++ - home",
   "text": 'Istanbul { Türkische Kampfflugzeuge haben an der Grenze zu Syrien ein ausländisches Luftfahrzeug abgeschossen.
       Bisher sei nicht bekannt, zu welchem Land das im türkischen Luftraum abgeschossene Objekt gehört habe, erklärte der türkische
       Generalstab auf seiner Internetseite. Aus der Mitteilung ging nicht hervor, ob es sich um ein
       Flugzeug, einen Hubschrauber oder eine Drohne handelte Trotz dreifacher Warnung habe das Luftfahrzeug seinen Kurs fortgesetzt
       und sei deshalb entsprechend der Einsatzregeln der türkischen Streitkräfte abgeschossen worden, erklärte der Generalstab.
       In den vergangenen Wochen waren russische Kampfjets bei Einsätzen über Syrien mehrmals in den türkischen Luftraum eingedrungen.
       Ankara warnte Moskau daraufhin, bei einer Wiederholung werde das Feuer eröffnet. Am Donnerstag war eine Delegation
       des russischen Militärs zu Gesprächen mit türkischen Offizieren in Ankara eingetroffen. Die türkische Luftwaffe hatte
       in den vergangenen Jahren ein syrisches Kampfflugzeug, einen Hubschrauber und eine Drohne abgeschossen. Mehr gleich bei BILD.de
       Teilen Twittern Teilen auf Google+ Teilen auf Tumblr Teilen auf Pinterest per Mail versenden Artikel melden'

}
```

Figure 3.2: Sample JSON title, with very little information [34]

3.3 News commentary

News Commentary[22] dataset provides parallel corpus containing news articles in various languages such as English, Dutch and German. Each article consists of a text body and a corresponding headline, each line has a corresponding translation to the other available languages. This makes this dataset capable of serving as training set for both translation and summarization. They are provided by WMT[35] for machine translation modeling. They are simply downloaded from the resource and processed line by line using a trivial text file reader object. The original text and their corresponding translations are stored in *list* objects, which is used as x and y components of the training set. This corpus contains parallel articles and headlines, however it was not used as summarization training set, because its size and format suits more towards translation model rather than a summarization model(which needs a much bigger dataset to start learning to generate meaningful sentences).

Language	Examples	Average Tokens
German	52020	23
Dutch	57906	25
English	57906	22

Table 3.3: News Commentary Dataset

Chapter 4

Approaches

4.1 Preprocessing

SentencePiece model is trained while using the ”byte-pair-encoding” subword algorithm for tokenization by passing the option for model type as *bpe*. This tokenization model’s independence from any particular language makes it a fitting candidate to be the tokenizing algorithm for the models implemented due to their multi-linguistic purpose. The training input used for this model is simply the text file from the summarization training set. The model is trained using the vocabulary size of 32000. This model is used later to encode the inputs and outputs into an embedding before passing on to the encoder and decoder side. The tokens are converted into individual ids to construct a training set for neural networks, the ids generated as outputs by the neural network can be converted back to tokens and sentences by referring to a vocabulary object.

```
Sentence = This is a good test
Tokens(Sentence, vocab size, 32000) =
                    [’This’,’a’,’good’,’test’]
Tokens(Sentence, vocab size, 12000) =
                    [’Th’,’is’,’a’,’g’,’ood’,’test’]
```

Figure 4.1: SentencePiece BPE Tokenization

4.2 Summarization and translation model

The transformer model described in 2.6 have proven to be state-of-the-art models for machine translation and automatic summarization tasks. This research aimed to utilize these properties to evaluate performances over dataset with multiple languages (English, German and Dutch). The architecture that was used for implementing transformers has been explained in section 2.6. Encoder and Decoder architecture has been implemented by using the *pytorch* modules which is available for "python" development, it is also available in *tensorflow*, however due to compatibility issues it "pytorch" was later preferred. Each of the stacks are supplied with 6 layers of encoder and decoders each, the model parameters were borrowed from the default parameters described in 2.6. During training the transformer encoder stack needs an embedded input. It's done by first tokenizing all the training sentences, followed by splitting the training set into batches. The training cycle works in batches, due to the uneven length of sentences across the corpus, a padding is applied to append a *pad index* at the trailing end of sentences to make all the length in the batch consistent. However, on a larger dataset the padding can also take a long time just during pre-propcessing phase, which is why the sentences are sorted by length and then padded to optimize training intervals. The batches are shuffled around after sorting to even the training iterations over an epoch. There were total of 14633 batches of 30 examples each across translation and summarization dataset combined. The decoder stack's final output at the end is processed further with beam search[28] decoder to extract most likely sentences that from the overall output based on the probabilities.

4.3 Cascaded approach

The primary objective of this research was to evaluate a trained summarization model's performance over a multi-lingual dataset to generate summaries specifically in other languages than English. Since the summarization training set is already and English to English mapping, it was established as a baseline to investigate the knowledge transfer across languages(English, German and Dutch) during automated summary generation.

In order to achieve such results one of the approaches implemented was a cascaded approach to multi-lingual summaries. It is a simple linear approach to automatic summarization in multiple languages through various summarization and translation models that were trained individually. It aims to streamline multiple text transduction models (transformers) together. These individual models are stacked together to process an input sentence to translate, summarize and then translate back in that order shown in 4.2. The original texts are passed through a series of transformer models (translation and summarization) to generate summary in a non-English language at the end of all transformer stack.

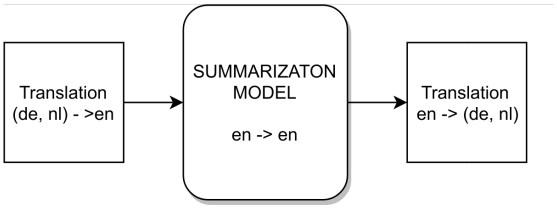


Figure 4.2: Cascaded Approach Architecture

4.4 Multitask Learning

Cascaded approach discussed in the previous section makes use of multiple models to generate summaries. Although it is a simple approach, it leaves room for discussion for the approach where multilingual summaries can be generated using a single model instead of multiple models working together. Another argument why a single multipurpose model is needed is that training multiple models takes more time and is computationally expensive. Additionally, each text transduction process consumes time independently. In order to address this efficiently, a case is made for proposing a multitask learning model, which can generate multi-lingual summaries.

Multi-task learning is a training method in which a machine learning model is trained to perform for multiple tasks at the same time. This method tries to exploit the similarities of the different tasks that are being solved at the same time [19]. Multitask learning approach works because when training for more than one task, a model can learn a more generalized representation of the tasks and there is a lower risk of over-fitting.

Pertinent to automatic summarization in multiple languages, a modified transformer model was implemented to accommodate the multi-tasking properties within the architecture of the model. The intuition behind this approach was that it should be possible to exploit the performance of transformer models in translation and summarization tasks by training the transformer model to do both tasks at the same time. The model was expected to learn a rather efficient representation of both tasks while also learning the patterns in the input sequence. However, the model needs a context to generate the appropriate text according to the input instructions (whether translating or summarizing). Inspired by the model proposed by [Johnson et al., 2017], this was solved by encoding a special set of tokens which define the task in progress (translation or summarization)[12]. The idea behind this approach is to encode the tasks within the input sequences as prefix user defined symbols. This is accomplished by modifying the way transformer model's input embeddings are processed at each time step.

```
Input Sequence X = [x_1, x_2, ..., x_N]
Task Enc(X1) = [SUMM, EN, EN, x_1, ..., x_N]
```

Figure 4.3: Multitask task encoding

Figure 4.3 shows how the training set examples are modified before training. This process modifies the input embedding supplied to the encoder stack in transformers, and the encoder outputs are now affected by the information (encoded tasks symbols), forcing the model to learn the pattern between the user defined task symbols and the expected output.

In the first iteration of the multitask learning model, the source sequences in training set are tagged with encoded instructions such as *[SUMM, TRAN, ...]* while the target sequences are left unchanged. The model's training set has been selected and filtered, so that the input sequences X are only in English language rather than extending across all three languages (English, German and Dutch) in the overall training dataset. This implied the first language token in all the encoded task sequences will be *EN*(English). It should be noted that the source sequences are always in English. The motivation behind this iteration of multitask learning model was to investigate the performance of the multitask model and if they differ due to the diversity of language domains in the training set.

Over the training period, the model is expected to learn the relation between the expected text and the encoded tasks. To whether to translate and summarize at any given time. This technique is implemented to make use of the fact that the training set for summarization, has been in English exclusively, so if a model can learn to do both separately it should be possible to force the model to do translation and summarization at the same time. In order to investigate the transfer of summarization representation across multiple languages, the initial format of the encoding can be manipulated to mix the instruction values together. Figure 4.4 shows an example of an encoded task which is asking the model to take a text in German(DE) and generated a summary(SUMM) of that text in German(DE). The test dataset constructed from *webhose* is used to evaluate the model's performance in multi-lingual automatic summarization.

```
Task Enc(X) = [SUMM, DE, DE, x_1, x_2, ...... x_N]
```

Figure 4.4: Encoded tasks to generate German summary of a German text

The training set for this task is a combination of summarization training set and the translation dataset where each input sequence from the training set is supplied to pre-processing component (shown in figure 4.5), which converts the sequences to their corresponding embedding however, they are prefixed by the encoded task symbols to indicate the desired output (translation or summarization).

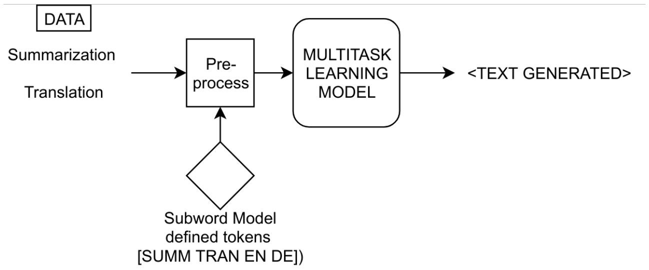


Figure 4.5: Multitask Learning Model

4.5 Multitask learning, multiple languages

The model presented in the previous section only has only trained on English inputs, however in order to make the model more efficient at summarizing non-English texts into either of the three languages(English, Dutch, and German) there is a need for model to also train on non English source input sequences. This is expected in order to force the model to learn the relationship between language symbols and summaries. Augmenting the model proposed in section 4.4, the complexity of the multitask learning is increased by extending the training set across multiple languages. Meaning, the input sequences can contain all three language tags (EN, DE, and NL). However, it should be noted that the only summarization data available is exclusively in English. This model is expected to learn the representation multilingual summarization due to availability of all languages within the input tags.

```
Source (X_1) = [SUMM, EN, EN, X_1]
Source (X_2) = [TRAN, EN, DE, X_2]
Source (X_2) = [TRAN, DE, EN, X_2]
Target (Y) = [Y]
```

Figure 4.6: Multitask learning Model, multiple input languages

The transformer model predicts next word based strictly on the words it has already predicted or seen so far, it should be possible to augment the set of considered words for prediction with the encoded task tokens. *SUMM* and *TRAN* are the token for summarization and translation tasks respectively. Similarly, the tokens for the input and output text's languages are also encoded within the sequence. For a translation training example X, which lets say is an input for translating an English text to German, the corresponding task can be encoded with X as *[TRAN EN DE X]*. These special tokens are assigned as user-defined symbols within the *SentencePiece* model.

4.6 Multitask Learning with tagged outputs

Section 4.5 discusses the modification of input embedding to force the model to learn the relation between the encoded task symbols and the text itself to generate corresponding outputs. For the next iteration of the multitask learning, this research aims to investigate the effect on the summaries generated through the modification on the decoder side. In the previous section, the model is provided with a context at the encoder side regarding the task through encoded task symbols. However, this model also aims to force the decoder to consider the task symbols separately during text generation along with the output from the encoder side. Transformer models generate texts by predicting words based on the already predicted words at each time step. While predicting a word every time step it is imperative that decoder mustn't lose context of the encoded tasks. The encoded tasks shown in figure 4.4 are inserted within each time step of decoder to force the model to learn the tasks based on instructions that came through the original input sequence. On the decoder side, the prediction is now forced to be dependent exclusively on the words that has been already predicted and the task symbols.

```
Source (X_1) = [SUMM, EN, EN, X_1]
Source (X_2) = [TRAN, EN, DE, X_2]
Target (Y) = [SUMM EN EN Y] \\ tagged
```

Figure 4.7: Multitask learning with, tagged target sequences

Figure 4.7 shows how the input and output sequences are modified to force the decoder to predict words based on the tasks assigned.

```
For, time step = t, Predict([SUMM, EN, EN, x_1, x_2, .....])
```

Figure 4.8: Multitask task encoding

The motivation behind implementing variations models described from section 4.5 to 4.6, was to evaluate the performances of the multi-task learning method based on the language domain in the training set and how the tagging affects the generated text's quality. Additionally also identifying the effects of tagging input and target sequences and whether that helps in contextualizing the text better for the summarization model during text generation.

Chapter 5

Evaluation Methods

5.1 Summarization

An ideal evaluation scenario for a computer generated summary, a human input would be required to properly attest the integrity of new sentences formed and analyze whether the underlying meaning and information from the original content is preserved in the summary. However, in case where the summarization models are working on large datasets, it is not feasible to employ required amount of human resources, additionally for the sake of consistency in evaluation metrics, a more mathematical approach is suitable for quantitative analysis. One of such approaches is called ROUGE or Recall-Oriented Understudy for Gisting Evaluation[14]. It is one of the popular methods to evaluate and score the quality of automatic summarization in the field natural language processing. It provides scores by counting the longest overlapping sequences, words and n-grams between the summaries generated by the machine learning models and the expected summary provided beforehand. There are software packages available in different development platforms. In this research, the ROUGE scores are calculated by utilising the *PyRouge*[26] package which is available as a Python module.

5.1.1 Summaries generated in non-English Languages

Testing set (*webhose*) used for the cascaded and multi-task model only provides non parallel corpus in multiple languages. In order to evaluate the quality of summary, the summaries are generated in the original languages, for example: German summary for a German text while having never trained the model specifically for German to German summarization. The output and reference is scored using the simple ROUGE algorithm.

5.2 Translation

Evaluating machine translation results through human resources is both time-consuming and expensive. In order to evaluate the quality of the machine translated text, one of the popular algorithms in natural language processing is called BLEU or Bilingual Evaluation Understudy [23]. The underlying theme behind this technique is that the closer the translation is to the one provided by humans, the better score this algorithm returns. It is a popular technique due to its simplicity, effectiveness, correlation with human evaluation and the fact that its language independent.

The approach works by comparing the candidate translation generated by a translation model to a target translation candidate. It works by comparing matching n-grams between the target translation sequence and the generated sequence. This comparison doesn't consider the order of the words.

Chapter 6

Experiments and Observations

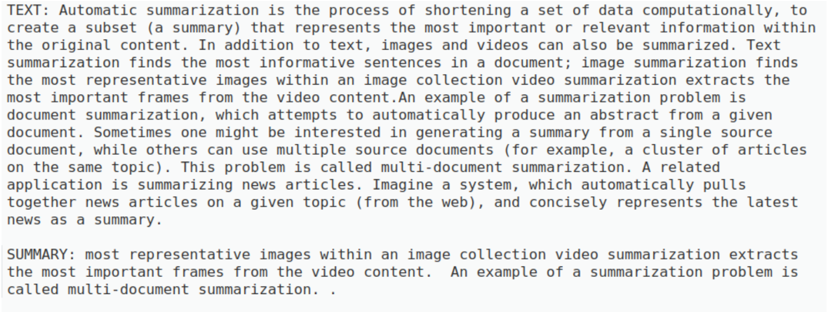
6.1 Summarization and translation model (News Commentary, CNN/DailyMail)

The transformer model[33] were designed from scratch using the *python* development tools and inbuilt modules. These models were trained on both *News Commentary* and *CNN/DailyMail* datasets for translation and summarization related tasks respectively. The corpus was split into testing and training sets separately to establish a measurement metric for the performances of combined models for translation and summarization tasks.

According to the results (see table 6.1), summarization models scored higher when evaluated on corpus containing multiple-sentences based summaries(*CNN/DailyMail*, while scoring comparatively lower on *News Commentary* corpus which has titles instead of summaries.

Training Dataset	Testing Dataset	ROUGE (R1) Score
CNN/DailyMail	CNN/DailyMail	24.01
CNN/DailyMail	Webhose	11.03
CNN/DailyMail	News Commentary	7.33

Table 6.1: Summarization model



```
TEXT: Automatic summarization is the process of shortening a set of data computationally, to
create a subset (a summary) that represents the most important or relevant information within
the original content. In addition to text, images and videos can also be summarized. Text
summarization finds the most informative sentences in a document; image summarization finds
the most representative images within an image collection video summarization extracts the
most important frames from the video content.An example of a summarization problem is
document summarization, which attempts to automatically produce an abstract from a given
document. Sometimes one might be interested in generating a summary from a single source
document, while others can use multiple source documents (for example, a cluster of articles
on the same topic). This problem is called multi-document summarization. A related
application is summarizing news articles. Imagine a system, which automatically pulls
together news articles on a given topic (from the web), and concisely represents the latest
news as a summary.

SUMMARY: most representative images within an image collection video summarization extracts
the most important frames from the video content.  An example of a summarization problem is
called multi-document summarization. .
```

Figure 6.1: Summarization sample

The translation model was trained on *News Commentary* dataset, as it provides a range of parallel corpus in multiple languages. The quality of the translation generated by the transformer models were evaluated by using BLEU[23] scores (see section 5). Table 6.2 details the evaluation results after training for 3.5 days on a GPU accelerated unit for each pair of languages.

Dataset	Languages	BLEU Score
News Commentary	English, German	29.13
News Commentary	German, English	28.71
News Commentary	English, Dutch	31.11
News Commentary	Dutch, English	31.23

Table 6.2: Translation model

6.2 Cascaded approach

First a summarization model is trained to summarize English texts into English summaries using the summarization dataset available (CNN/DailyMail) [4]. This serves as the central component in this approaches' architecture as shown in figure 4.2. Since the summarization model can only summarize English sentences, it is facilitated by various translation models. These translation model first, translate text from German or Dutch to English (making it compatible for the summarization model). The generated summary (English text) is followed by another translation model to eventually generate a summary in German or Dutch. It is worth noting that all the implemented models are transformer models as explained in section 2.6.

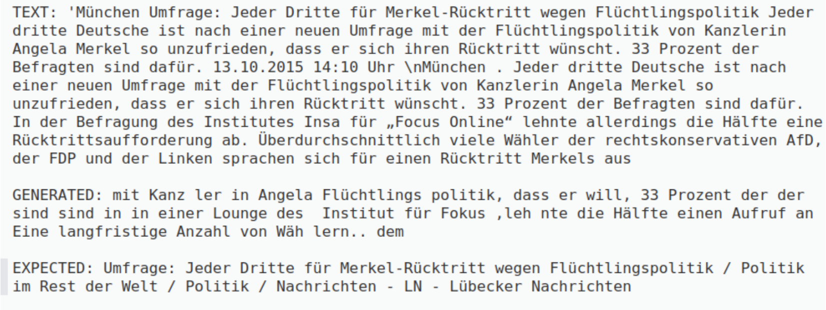
Webhose provides independently non parallel news article dataset in English, German and Dutch languages. That is used to first translate the text to English, and then summarize and translate it back to the original language. The overall output can be compared against the original expected summary and evaluated using methods that were described in chapter 5. It has been observed as showed in figure 3.2 that sometimes the ground truth titles or summaries are just meta-

Dataset	Languages	ROUGE(R1) Score
Webhose	German	9.80
Webhose	Dutch	9.62

Table 6.3: Cascaded approach scores

data about the text rather than actually having a meaningful content, or just noisy data. Additionally, since the text is passed through multiple models, the sentences seem to lose on quality over the period due to the noise added from particular models. It hurts the quality of the output at the end because the final output text ends up containing words which are not pertinent to the original text. It can be seen that despite, the low scores the ROUGE(R1) scores for multi-lingual summarization is similar to whats been described in section 6.1. The low scores can be reasoned with the lack of available common n-grams when the reference summaries (in this case titles) are just a small sentence consisting of very few words.

The overall streamlined system generates summaries in two languages, English and German or Dutch. The ROUGE(R1) scores are calculated by referring to the original title in the original language. For example, to examine the multi-lingual knowledge transfer for German language, according to the dataset available in this research, the German text content is first translated into English and then it's summarized which is followed by another step of translation to German again. After this series of steps is complete the result is a summary generated in German, hence it is comparable to the original titles which were in German for evaluation. Table 6.3 show the various ROUGE scores achieved using the cascaded approach. Each individual models took up to 3 days while being executed on a GPU accelerated machine.



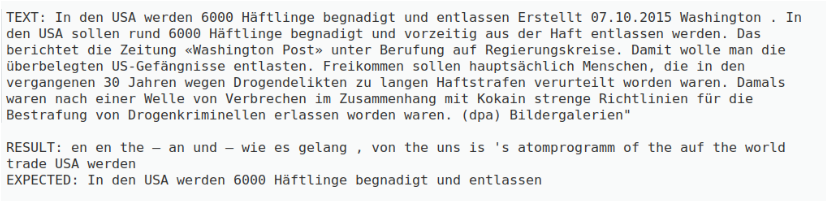
```
TEXT: 'München Umfrage: Jeder Dritte für Merkel-Rücktritt wegen Flüchtlingspolitik Jeder
dritte Deutsche ist nach einer neuen Umfrage mit der Flüchtlingspolitik von Kanzlerin
Angela Merkel so unzufrieden, dass er sich ihren Rücktritt wünscht. 33 Prozent der
Befragten sind dafür. 13.10.2015 14:10 Uhr \nMünchen . Jeder dritte Deutsche ist nach
einer neuen Umfrage mit der Flüchtlingspolitik von Kanzlerin Angela Merkel so
unzufrieden, dass er sich ihren Rücktritt wünscht. 33 Prozent der Befragten sind dafür.
In der Befragung des Institutes Insa für „Focus Online“ lehnte allerdings die Hälfte eine
Rücktrittsaufforderung ab. Überdurchschnittlich viele Wähler der rechtskonservativen AfD,
der FDP und der Linken sprachen sich für einen Rücktritt Merkels aus

GENERATED: mit Kanz ler in Angela Flüchtlings politik, dass er will, 33 Prozent der der
sind sind in in einer Lounge des  Institut für Fokus ,leh nte die Hälfte einen Aufruf an
Eine langfristige Anzahl von Wäh lern.. dem

EXPECTED: Umfrage: Jeder Dritte für Merkel-Rücktritt wegen Flüchtlingspolitik / Politik
im Rest der Welt / Politik / Nachrichten - LN - Lübecker Nachrichten
```

Figure 6.2: Cascaded approach results

Figure 6.2 shows a summarization generated through a cascaded model for an original German text. Even though the output summary does contain pertinent words or phrases, in its entirety the summary is not making a semantic sense. These type of results are observed sporadically throughout the testing set.

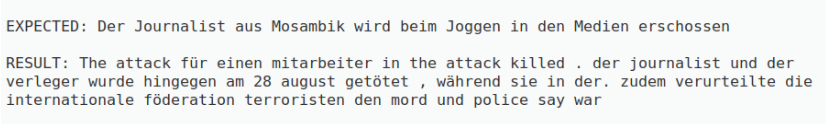
6.3 Multitask Learning with only English inputs

This is the model discussed in section 4.4 where the training set data only contains input sequences in English, and the target sequence could extend across multiple languages (explained in section 4.4). It was observed that, this method does only produce results in German/Dutch, as instructed and expected from the input sequences. Despite the summaries generated are not of the highest quality, it still contains the phrases and words from the expected summaries (see figure 6.3 and 6.4). This resulted in an evaluation score of ROUGE(R1) = 2.54.



```
TEXT: In den USA werden 6000 Häftlinge begnadigt und entlassen Erstellt 07.10.2015 Washington . In
den USA sollen rund 6000 Häftlinge begnadigt und vorzeitig aus der Haft entlassen werden. Das
berichtet die Zeitung «Washington Post» unter Berufung auf Regierungskreise. Damit wolle man die
überbelegten US-Gefängnisse entlasten. Freikommen sollen hauptsächlich Menschen, die in den
vergangenen 30 Jahren wegen Drogendelikten zu langen Haftstrafen verurteilt worden waren. Damals
waren nach einer Welle von Verbrechen im Zusammenhang mit Kokain strenge Richtlinien für die
Bestrafung von Drogenkriminellen erlassen worden waren. (dpa) Bildergalerien"

RESULT: en en the – an und – wie es gelang , von the uns is 's atomprogramm of the auf the world
trade USA werden
EXPECTED: In den USA werden 6000 Häftlinge begnadigt und entlassen
```

Figure 6.3: Multitask model sample results, only English input sequences

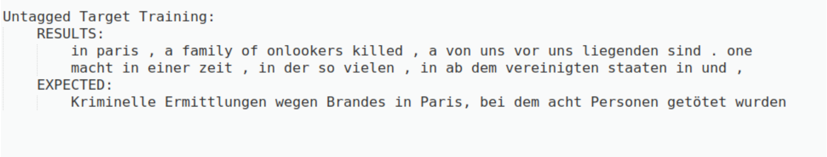


```
EXPECTED: Der Journalist aus Mosambik wird beim Joggen in den Medien erschossen

RESULT: The attack für einen mitarbeiter in the attack killed . der journalist und der
verleger wurde hingegen am 28 august getötet , während sie in der. zudem verurteilte die
internationale föderation terroristen den mord und police say war
```

Figure 6.4: Multitask model sample results, only English input sequences

6.4 Multitask Learning Model, multiple languages

When the model was trained without the encoded tags (*[SUMM, EN, DE, TRAN]*) in the target output sequences, the training progress and learning curve was similar to the other multi-task learning models. However, the results as an example shown in the figure 6.5 show that the results seem to contain words from multiple languages in the same sentence. It can be postulated that due to the training set extending across multiple languages domain, the model when instructed to generated summaries in another language, it attempts to combine both learned tasks [translation and summarization], however not producing the results in one language as instructed from the input sequence. So as expected from the sample outputs, the ROUGE(R1) scores obtained on German dataset was 1.14.

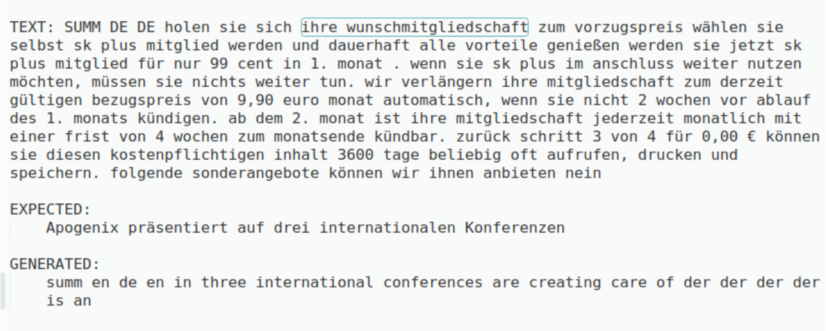


```
Untagged Target Training:
    RESULTS:
        in paris , a family of onlookers killed , a von uns vor uns liegenden sind . one
        macht in einer zeit , in der so vielen , in ab dem vereinigten staaten in und ,
    EXPECTED:
        Kriminelle Ermittlungen wegen Brandes in Paris, bei dem acht Personen getötet wurden
```

Figure 6.5: Multitask model sample results, untagged outputs

6.5 Multitask learning model with tagged output

As detailed in section 4.6, the multi-task learning was implemented by modifying the encoder-decoder input embedding mechanism to consider the encoded task sequences within the input during text generation. After training the multitask model, it turns out that when the encoded sequences are manipulated to simultaneously perform the summarization and translation, the model ends up producing non-conclusive outputs, performing very poorly. However, during the training phase of the model, the learning curve seems to progress as expected however the results do not show good results, as shown in figure 6.6. While the output texts can be seen to have repeated the encoded symbols to what it was trained on(”English” to ”English” summarization) instead of generating the text. The outputs seemed to contain words from multiple languages as well, resulting in an average ROUGE(R1) score of 1.01 across on German dataset.

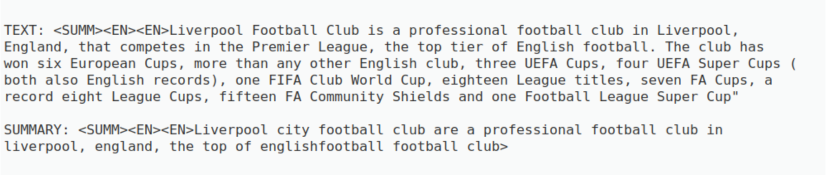


```
TEXT: SUMM DE DE holen sie sich ihre wunschmitgliedschaft zum vorzugspreis wählen sie
selbst sk plus mitglied werden und dauerhaft alle vorteile genießen werden sie jetzt sk
plus mitglied für nur 99 cent in 1. monat . wenn sie sk plus im anschluss weiter nutzen
möchten, müssen sie nichts weiter tun. wir verlängern ihre mitgliedschaft zum derzeit
gültigen bezugspreis von 9,90 euro monat automatisch, wenn sie nicht 2 wochen vor ablauf
des 1. monats kündigen. ab dem 2. monat ist ihre mitgliedschaft jederzeit monatlich mit
einer frist von 4 wochen zum monatsende kündbar. zurück schritt 3 von 4 für 0,00 € können
sie diesen kostenpflichtigen inhalt 3600 tage beliebig oft aufrufen, drucken und
speichern. folgende sonderangebote können wir ihnen anbieten nein

EXPECTED:
    Apogenix präsentiert auf drei internationalen Konferenzen

GENERATED:
    summ en de en in three international conferences are creating care of der der der der
    is an
```

Figure 6.6: Multitask model sample results

It can also be observed that the multitask learning model suffers from exposure bias. It has only learned to summarize when instructions are *”SUMM EN EN”*. So even though the input sequence starts from ”*SUMM DE DE*”, the summary generated repeats ”EN” after the tags that were originally provided.

In order to investigate the above observations, multiple variations of training set were trained with the model. However, in every iteration the model performs well when the training set only deals with either translation (one way) or summarization, but not both at the same time. Figure 6.7 shows the result of the multi-task model while trained only for summarization. Similar, results can be seen for translation as well.



```
TEXT: <SUMM><EN><EN>Liverpool Football Club is a professional football club in Liverpool,
England, that competes in the Premier League, the top tier of English football. The club has
won six European Cups, more than any other English club, three UEFA Cups, four UEFA Super Cups (
both also English records), one FIFA Club World Cup, eighteen League titles, seven FA Cups, a
record eight League Cups, fifteen FA Community Shields and one Football League Super Cup"

SUMMARY: <SUMM><EN><EN>Liverpool city football club are a professional football club in
liverpool, england, the top of englishfootball football club>
```

Figure 6.7: Multitask model sample results, English summary

6.6 Overview of multitask learning approaches

As it can be observed from the results discussion in section 6.3 to 6.5, the multitask learning performs relatively better when the training input domain is constrained within a singular language (English in this research's case). While in the later iterations with tagging the output sequences on the decoder side didn't seem to make a lot of difference in terms of evaluation scores (see table 6.4). The next iteration containing multiple language (English, German and, Dutch) was evaluated at lower ROUGE(R1) scores than the previous model(singular language inputs). These lower scores are mainly due to the fact that the output generated sequences contained words from multiple languages, and were not following the expected language domain according to the encoded sequences ("EN", "DE", or "NL"). The next iteration of the model containing multiple languages and tagged output sequences, performed similarly at lower scores (refer table 6.4). The summaries generated contained repeated words and phrases while also containing mixed words from different languages, resulting in a lower evaluation scores.

Multitask Learning Approach	ROUGE(R1) Score
English inputs only	2.54
Inputs with multiple languages	1.14
English input only with tagged output	2.21
Tagged output sequences, multiple languages	1.01

Table 6.4: Multitask Learning model evaluation over webhose

The multitask learning model works well, and produces more consistent texts as long as the training input sequences do not extend to multiple languages, once the training domain extends across other languages the texts generated start containing repeating phrases and mixed words from multiple languages.

6.7 Training observations and parameters

Each of the models (translation and summarization) are based on the transformer models. The starting training parameters were defined based on the availability of resources and the state-of-the-art translation and summarization model[33] (see more at table 6.5). Encoder and Decoder sides contain six layers each, while every layer containing 2 sublayers which are followed by dropout[29], produce an output of dimensionality 512. The dropout is applied for regularization[29]. Each self attention layer is equipped with 8 parallel heads, which computes the attention values in parallel.

Parameters	Values
Encoder Layers	6
Decoder Layers	6
Batch Size	1500
α	0.0001
Training steps	50000
Dropout	0.1
Attention Heads	8
d_{model}	512

Table 6.5: Model parameters

Training the multitask learning model took the highest amount of time due to it's bigger in comparison dataset (since it contains combination of both summarization and translation dataset).

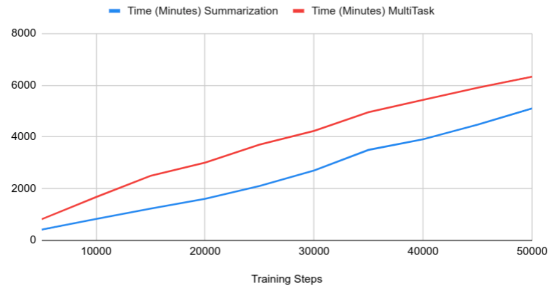


Figure 6.8: Multitask and Summarization Model: Training Time

Default starting values for learning rate(α) is set to 0.00001 for all the models, the learning curve over the training steps for summarization and multitask model can be seen in figure 6.9.

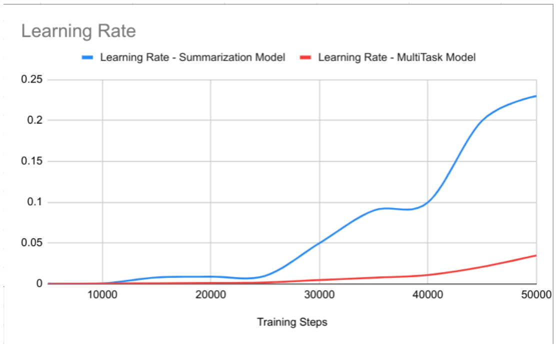


Figure 6.9: Multitask and Summarization Model: Learning Curve

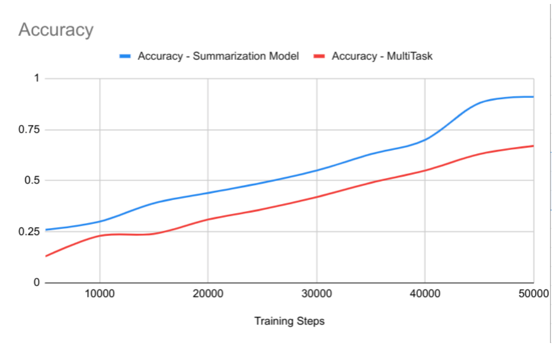


Figure 6.10: Multitask and Summarization Model: Accuracy

It can be seen that clearly the multitask learning model reaches higher accuracy scores and higher learning rates over the training steps. While they are similar architectures, they do differ in the size of the dataset which also factors in towards the difference in learning curve.

Chapter 7

Summary

This research explored various approaches to achieve automatic summarization across multiple languages. While drawing inspiration from state-of-the-art summarization and translation models [33][20][12]. Based on the findings and proposed models an encoder-decoder architecture was employed using transformer models[33]. The translation and summarization training set were drawn from *News Commentary* and *CNN/DailyMail* [4][22]. The testing dataset, however, was fetched from *webhose*[34]. *Webhose* was preferred because it provided freely available formatted dataset in multiple languages (English, Dutch and German).

Summarization model individually was evaluated at ROUGE(R1) score of 24.01 on *CNN/DailyMail* dataset. However, when tested on *webhose* dataset, they performed worse at ROUGE(R1) score of 11.03. The poor result was inspected, the reason behind the lower evaluation score is due to the fact that *webhose* contains a plenty of titles with no pertinent content or just metadata about the article (indicated before in figure 3.2). It was not feasible to go through the entire corpus for checking the titles manually. Absence of longer summary also hurts the model's score due to the training set containing much more detailed summaries.

Translation model was trained on *News Commentary* dataset, as it was the only dataset available which provided parallel corpus in the languages pertinent to this research. Translation model was evaluated at an average BLEU score of 30.04 across English, German and Dutch languages.

First, a cascaded approach is proposed to achieve multilingual automatic summarization through the means of combining the aforementioned summarization and translation models. It was observed that it is possible to generate multi-lingual summaries through the means of processing texts through various models serially, the end result can be evaluated by referencing the expected output. However, due to multiple phases of text transduction, a noise is added to the text after each step, which factors in at the end when evaluating the quality of the summary using methods such as ROUGE(R1) scores, which were evaluated at 9.70 in average for all three languages. It should also be noted that the quality of the summaries/titles available in *webhose* corpus detriments the results as well.

Subsequently a multitask learning model was proposed to investigate whether a model can learn summarization and translation tasks separately, and whether it can generated readable summaries while only having being trained on English-to-English summaries, English-to-German translation and German to English translation. The underlying idea behind the approach was inspired from the proposed mode by [Johnson et al., 2017]. The model learned summarization and translation at the same times, while the input sequences were encoded with special symbols to indicate the current task (translation or summarization). The training of multitask model took the longest due to much bigger size of the dataset. When evaluated, it was observed that the outputs were not generating readable summaries, and generated tokens and phrases were getting repeated. Multiple variations (described in section 4.4 to 4.6) of multi-task model were trained to investigate the challenges and the nature of text generated. When the input and target sequences are encoded with the tasks, the model seems to suffer from exposure bias, because during generation it relies on generated encoded task symbols. The generated summaries appeared to contain words and phrases from multiple languages within the same output. Results were slightly better for models which were trained with a singular language data (input sequences only in English). This generated slightly better results than the other iterations of the multitask learning model, it was more consistent to stay within the desired language domain and contained some pertinent phrases and words.

Bibliography

[1] Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer Normalization". In: *ArXiv* abs/1607.06450 (2016).

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate.* cite arxiv:1409.0473Comment: Accepted at ICLR 2015 as oral presentation. 2014. URL: `http://arxiv.org/abs/1409.0473`.

[3] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. URL: `http://www.aclweb.org/anthology/D14-1179`.

[4] *CNN/DailyMail.* URL: `https://github.com/JafferWilson/Process-Data-of-CNN-DailyMail`.

[5] Dipanjan Das and André F. T. Martins. *A Survey on Automatic Text Summarization.* 2007.

[6] *Extraction Based Summarization.* URL: `https://blog.floydhub.com/gentle-introduction-to-text-summarization-in-machine-learning/`.

[7] Jonas Gehring et al. "Convolutional Sequence to Sequence Learning". In: *Proceedings of the 34th International Conference on Machine Learning.* Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, 2017, pp. 1243–1252. URL: `http://proceedings.mlr.press/v70/gehring17a.html`.

[8] Sebastian Gehrmann, Yuntian Deng, and Alexander Rush. "Bottom-Up Abstractive Summarization". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.* Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 4098–4109. DOI: `10.18653/v1/D18-1443`. URL: `https://www.aclweb.org/anthology/D18-1443`.

[9] Makoto Hirohata et al. "Sentence extraction-based presentation summarization techniques and evaluation metrics". In: *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on* 1 (Jan. 2005). DOI: `10.1109/ICASSP.2005.1415301`.

[10] Sepp Hochreiter. "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6 (Apr. 1998), pp. 107–116. DOI: `10.1142/S0218488598000094`.

[11] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[12] Melvin Johnson et al. "Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation". In: *Transactions of the Association for Computational Linguistics* 5 (2017), pp. 339–351. DOI: `10.1162/tacl_a_00065`. URL: `https://www.aclweb.org/anthology/Q17-1024`.

[13] C. -. Lee et al. "Word recognition using whole word and subword models". In: *International Conference on Acoustics, Speech, and Signal Processing,* 1989, 683–686 vol.1. DOI: `10.1109/ICASSP.1989.266519`.

[14] Chin-Yew Lin. "ROUGE: A Package for Automatic Evaluation of summaries". In: *Proc. ACL workshop on Text Summarization Branches Out.* 2004, p. 10. URL: `http://research.microsoft.com/~cyl/download/papers/WAS2004.pdf`.

[15] Linqing Liu et al. "Generative Adversarial Network for Abstractive Text Summarization". In: (Nov. 2017).

[16] I Mani. *Automatic Summarization.* John Benjamins Publishing Company, 2001.

[17] Tomas Mikolov et al. "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in Neural Information Processing Systems 26.* Ed. by C. J. C. Burges et al. Curran Associates, Inc., 2013, pp. 3111–3119. URL: `http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf`.

[18] Tomas Mikolov et al. "Recurrent neural network based language model." In: *INTERSPEECH.* Ed. by Takao Kobayashi, Keikichi Hirose, and Satoshi Nakamura. ISCA, 2010, pp. 1045–1048. URL: `http://dblp.uni-trier.de/db/conf/interspeech/interspeech2010.html#MikolovKBCK10`.

[19] Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, Lukasz Kaiser. "Multi-task Sequence to Sequence Learning". In: (2015).

[20] Ramesh Nallapati et al. "Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond". In: *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning.* Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 280–290. DOI: `10.18653/v1/K16-1028`. URL: `https://www.aclweb.org/anthology/K16-1028`.

[21] Joel Larocca Neto et al. *Document Clustering and Text Summarization.* 2000.

[22] *News Commentary Dataset.* URL: https://www.statmt.org/wmt19/.

[23] Kishore Papineni et al. "Bleu: a Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics.* Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, July 2002, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://www.aclweb.org/anthology/P02-1040.

[24] Romain Paulus, Caiming Xiong, and Richard Socher. "A Deep Reinforced Model for Abstractive Summarization". In: *International Conference on Learning Representations.* 2018. URL: https://openreview.net/forum?id=HkAClQgA-.

[25] Alec Radford, Luke Metz, and Soumith Chintala. *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks.* cite arxiv:1511.06434Comment: Under review as a conference paper at ICLR 2016. 2015. URL: http://arxiv.org/abs/1511.06434.

[26] *ROGUE Package.* URL: https://github.com/bheinzerling/pyrouge.

[27] Abigail See, Peter J. Liu, and Christopher D. Manning. "Get To The Point: Summarization with Pointer-Generator Networks". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* Vancouver, Canada: Association for Computational Linguistics, July 2017, pp. 1073–1083. DOI: 10.18653/v1/P17-1099. URL: https://www.aclweb.org/anthology/P17-1099.

[28] Yusuke Shibata et al. "Byte Pair Encoding: A Text Compression Scheme That Accelerates Pattern Matching". In: (Sept. 1999).

[29] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

[30] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. "LSTM Neural Networks for Language Modeling". In: *INTERSPEECH.* 2012.

[31] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to Sequence Learning with Neural Networks". In: *Advances in Neural Information Processing Systems 27.* 2014.

[32] *Tensorflow CNN Daily Mail.* URL: https://www.tensorflow.org/datasets/catalog/cnn_dailymail.

[33] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30.* Ed. by I. Guyon et al. Curran Associates, Inc., 2017, pp. 5998–6008. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

[34] *Webhose.* URL: http://webhose.io.

[35] *WMT*. URL: https://www.statmt.org/.