Cross-Lingual Text Summarization

The purpose of this report is to explore and implement transformer-based summarizer natural language processing models for the purpose of condensing document text from a source language into a shortened version written in a target language.

**Introduction**:

Natural language processing can be defined as the analysis and creation of natural language via computational means. In this scenario, the implementation of natural language processing is required to understand text so that it can be summarized accurately (pertaining to content) and in a way that produces output that is understandable in the target language.

A transformer is model is a deep learning model architecture that utilizes a “self-attention” mechanism which allows the weight manipulation of model inputs relative to their determined importance. Through effect, the model learns context and relationships between sequential data – a form of statistical understanding (Merrit & NVIDIA, 2022).

Document or text summarization is simply the processing of input into an output version shorter and more concise.

**Functionality**:

Computers, and thus computational models, cannot directly process language, so the input data must be first translated into numerical data which the learning model can utilize. The problem with this being that there are infinite ways to represent any given language data, text, or even letter.

There are many encoding schemes through which text can be converted into numbers: One-hot encoding, integer word representations, and dense vector-based word representations. The main problems with the first 2 schemes being the lack of relation between individual words, and the sheer number of potential words. Dense vector-based word representations are computed in a vector space, allowing for words with similar meanings to be close together, for this reason, making this scheme most appropriate for context dependent processing. An example of a text-vector encoding scheme is a co-occurrence-based word representation, where words are clustered in this continuous vector space based on how they are actually used in language.

A tokenizer is a function that splits given data into more general representations, such as from a sentence to a list of words. These ‘tokens’ are then converted into numbers, understandable by a computer (*Tokenizers - Hugging Face NLP Course*, n.d.).

This process of translating from high level languages such as human language, to low level, computer processible language, is called encoding. The reverse process is, as expected, called decoding. Due to the nature of these given models, an encoder model is best suited for tasks such as classification, understanding the meaning of text. A decoder model generates output sequences of data or text in this case. There also exists encoder-decoder models, also called sequence-to-sequence models, which have access to information pre encoding, allowing justification and greater context understanding of given texts. Such a model is best used for creating new text given input text, such as for summarization. An example of a representative of each an encoder, decoder, and encoder-decoder model is BERT, CTRL, and BART, respectively, although many such models exist. These functions all require certain tokenization methods to properly process data, therefore the tokenization is decided by the respective Transformer model. Due to the above reasoning, this report will use sequence-to-sequence modelling.

The models outlined above are all transformer-based models designed for different types of contextual mapping tasks and implement transformer-based learning to determine word representations, and therefore similarly encode text data numerically to the word-embedders above, only through different mechanisms.

Each of the mentioned models above are very large, and trained on extensive amount of data, the creation of which being a task that is incredibly expensive and time consuming. Due to this, along with the proven application validity of the models, for the course of this report some of these pre-trained models will be fine-tuned and utilized through transfer learning.

**Evaluation**:

To evaluate the proficiency and accuracy of a transformer model in such a natural language processing (NLP) task, the model’s summary output could be manually compared to what might be expected, however, this could take massive amount of time due to the number of samples required. Alternatively, pre-existing datasets can be used with both full text articles, and smaller summaries. The output of the model given the full text article would then be compared to the desired summary. The comparison can be computed through many different evaluation metrics, which could also, in turn, provide a loss value for the model to minimize. An example of an evaluation metric is ROUGE; while many variations exist, they generally function by calculating the overlap between generated and reference values based on co-occurrence statistics (Lin, 2004).

As this report’s tasks involve creating the ‘best’ model, this requires context of current models considered the ‘best’, the tasks are relative to the means of model evaluation. The deep-learning model and paper hub paperswithcode.com will be used to source reference models for these tasks. As the majority of, if not all, papers and models on the reference hub report performance metrics with ROUGE variation metrics, these metric results will be used to represent the model performance. Therefore, ROUGE metrics will be recorded for models created for this task, to be compared to those reference models. The ROUGE values recorded will encompass a few values to ensure validity over consistency. These metrics will be ROUGE-1, ROUGE-2, ROUGE-L. From these, the statistically oriented F1-score will be used to evaluate scores with respect to precision and accuracy. The ROUGE-L score is a function of the precision and recall of the longest common subsequence of correct prediction data.

The evaluation metric BLEU was also considered for use; however, BLEU does not consider structural information, or the order or coherence of words. While it certainly has some potential benefits over ROUGE, particularly in a translation scenario, it is ill-suited for summarization and therefore a higher BLEU score may not be indicative of a better summarization model.

To statistically compare the model performances, a t-test will be used as it is best for determining if there is a statistically significant difference between groups at the 95% confidence interval. If a model is shown to have an improvement over existing state-of-the-art models with t-test showing confidence over 95%, then it is shown that the model is better, at least in terms of the evaluation metric utilized.

**Dataset**:

The dataset to be used for this report is the WikiLingua corpus: a dataset of WikiHow article and summary pairs in 18 languages. All tests and training will be conducted on said dataset. This dataset specificity does mean however that while models function well on this dataset, results will not necessarily carry over to other datasets and contexts.

**Task 1**:

This task entails creating the best mono-lingual transformer-based summarizer model that can create an English summary of an English WikiHow article.

**Reference**:

The current best performing summarization model for the WikiHow lingua, according to the ROUGE metric statistics – for all variations, is presented by Savelieva et al. (2020). In this mentioned model, state-of-the-art results are achieved with evaluation metrics and values displayed below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | ROUGE-1 F1 | ROUGE-2 F1 | ROUGE-L F1 |
| **Value** | 35.91 | 13.9 | 34.82 |

These results were retrieved with the use of a pretrained BERT encoder and a randomly initialised transformer decoder. Interestingly, very few different models have been listed with state-of-the-art performance for summarization on the WikiHow dataset.

While this model performs excellently, I am unable to download it for fine-tuning, for this reason, this task will involve attempting to recreate similar results.

Despite the formula similarities between Rouge calculation functions, different library Rouge functions can produce different results for the same data, therefore the results shown in this report may differ from those in previously illustrated works of literature.

**Model**:

The first step of creating any model is data pre-processing. For this task, the WikiLingua dataset text and summary variables are extracted and put together to form a new, more accessible dataset. After this, the data needs to be tokenised so that it can be understood by the model, in the case of using pre-trained models, a pipeline function takes care of data pre-processing and tokenisation.

The first model functions to be determined are the data encoder and decoder – arguably the most important factors of a transformer model. These pretrained model blocks compile to build the complete model. Below the encoder-decoder (sequence-to-sequence) model is shown against F1 evaluation metric results. The performance results are averaged over 100 WikiHow dataset texts.

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric : Model** | ROUGE-1 F1 | ROUGE-2 F1 | ROUGE-L F1 |
| T5-Base | 20.12 | 4.91 | 15.13 |
| BART-Large | 17.95 | 5.00 | 10.58 |
| Pegasus-Large | 13.53 | 2.79 | 10.51 |

The process of conducting these tests involved generating summaries of the WikiHow documents and comparing these to the outputs with the actual WikiHow summaries corresponding to the document with ROUGE F1 metrics.

The previous testing was also conducted on a Bert-small 2 Bert-small summarizer PyTorch model, but only reached ROUGE-L F1 scores of around 10.78.

It is important to note that these models were trained on datasets other than the WikiHow lingua, therefore it is of no surprise that Rouge metric results barely approach that of the state-of-the-art model above.

Unfortunately, due to the sheer size of every version of the T5-Base model, I am unable to load the model outside of a direct pipeline for inference. On attempting to load in the model for fine-tuning, an OOM resource exhausted error is thrown. Due to this, and the sheer applicability of T5 models, the T5-small model will be experimented with for fine-tuning in an attempt to produce the best performing summarizer model on the WikiHow dataset.

**Fine-Tuning**:

For fine tuning, the dataset has been split into three sections: training, testing, and validation. The training set will be comprised of 90% of the original dataset, and the testing and validation datasets 5% each. The reasoning for including a validation dataset split being for contextual training information. The split sizes were determined through testing to ensure the training split be sizeable enough to adapt the model (must be significant as model is large and training data cannot be augmented to leverage training), while leaving enough data the model has not yet encountered for testing the model and training validation. The determined splits gave both model improvements, a valid testing set, and valuable training feedback.

After one epoch with otherwise base hyperparameters and a batch size of 8, the following results are recorded:

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | ROUGE-1 | ROUGE-2 | ROUGE-L |
| **Value** | 27.00 | 9.67 | 22.94 |

These results show a promising increase over the base T5-small model.

Conducting a t-test on the resultant ROUGE-L F1 scores from both the base and fine-tuned T5 models over 100 tests produces a t-value of 11.81. The corresponding p-value is < 0.001, meaning the result is significant at p < 0.05, providing evidence of statistical significance at the 95% confidence interval. It can therefore be concluded that this fine-tuned model performs better on summarization for the WikiHow dataset than the T5-Base model.

Increasing the number of epochs is shown to improve the Rouge scores for all metrics, unfortunately, due to computational capacity limitations, the number of epochs must be limited to only 3.

The series value in the table above denotes the epoch number corresponding to its metric results. An increase in epochs provides an overall increase in performance accuracy measures. Therefore, it is important to note that the model could potentially be trained for longer to reach greater performance, however, due to an inability to augment data, overfitting would eventually begin to occur.

Increasing the batch size over 8 results in the same out of memory issue as before and decreasing it below 8 reduces model performance according to ROUGE metrics – average difference of 0.1 units per metric with a batch size of 4 relative to that of size 8.

The learning rate value was shown to slow learning if too small and overtrain to the data if too large. The learning rate when set to 0.00002 showed to be the most consistent, avoiding both overtraining and slower than necessary learning. Altering the learning rate by even just 0.00001 showed effects of up to a 0.1 metric score difference after only one epoch.

The decay rate of the model optimizer was set at 0.01 as no or little decay resulted in over training, while too great a decay rate prevented the model from learning past few epochs.

Given that this is a summarization task, an input length of 1024 was set to accommodate for the majority of the data input, and the output length was limited to 128 to accommodate for the majority of the summary data.

The optimization function was chosen to be Adam weight decay as an adaptive algorithm can morph to the data, this is important as it isn’t strictly known what the model is to encode.

After training the final model, the following results were given:



While this model does not reproduce results found in the reference model, it certainly produces better results than the base model from which fine-tuning began. It could also be hypothesized that continued training on the WikiHow dataset would cause the model to approach the reference model further.

**Task 2**:

This task entails creating the best cross-lingual transformer-based summarizer model that can create a German summary of an English WikiHow article.

**Reference**:

Current models found in the Papers with Code library do not complete the task of summarization and translation, for this reason, reference testing will be conducted on combinations of these two model types. The models assessed will be multi-lingual as the designated task spans across languages.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Method : Metric** | Summarization | Translation | Summ – Trans | Trans – Summ |
| ROUGE-1 | 2.00 | 11.0 | 13.0 | 11.0 |
| ROUGE-2 | 0.00 | 1.00 | 1.00 | 1.00 |
| ROUGE-L | 2.00 | 7.00 | 9.00 | 7.00 |

The table above shows the ROUGE-L F1 scores of different combinations of model type. The models used were the “google/mt5-base” model for summarization, and the “Helsinki-NLP/opus-mt-en-de” model for translation. The testing input data was the English version of WikiHow articles, and the output data used for reference was the corresponding German version of the WikiHow article. This data was provided by the GEM/Wiki\_Lingua dataset discussed below. Testing was conducted over 100 input and reference data pairs. The stated multi-lingual models utilized were chosen as they are well backed with good reputation and are believed to be consistently valid models for their respective task, as well as this, they are all capable of processing multi-lingual values. Surprisingly, the results show little to no improvement introducing a summarization model to translated model output.

The results of this testing show that, even in combination, these models struggle to achieve considerable Rouge metric scores, with the best combination; Summarization followed by Translation, achieving an average ROUGE-L score of only 9.00. Importantly, these results also show that simply utilizing a translation model achieves high metric result values and would therefore be a good base model for the purpose of transfer learning: adapting an existing model to better fit a given task. This lack of significant sequential model performance improvement suggests that utilizing multiple models is perhaps not the most apt method of cross-lingual summarization.

**Model**:

The dataset originates from GEM/Wiki\_Lingua of HuggingFace datasets. This dataset maps cross-lingual versions of WikiHow data entries, the English to German split was downloaded. Due to the sheer amount of data (~40,000 pairs), it is assumed that utilizing only the training split predetermined is necessary. For reasons similar to those described in the first task, the retrieved dataset will be split into training, testing, and validation data, with the same split percentages. Excess dataset columns will be removed for simplicity’s sake as well as processing efficiency.

While this task could easily be broken down into its simpler functional components of translation and summarization, utilizing multiple functions/models introduces more room and for error to be propagated through the models, limiting the effective learning potential – as is shown through the combination of multiple models in sequence achieving minimally greater performance on the cross-lingual summarization above.

While the above testing suggests that a translation model may be a better base model for transfer learning, both summarization and translation models will be tested to determine the best model.

Below the sequence-to-sequence model is shown against F1 evaluation metric results. The performance results are averaged over 100 WikiHow dataset English texts to German summary pairs.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model Type :** | **Summarization** | | **Translation** | |
| **Model :** | MT-5 Base | MBART Large | Helinski-NLP | WMT19 |
| ROUGE-1 | 2.00 | 2.00 | 11.0 | 16.0 |
| ROUGE-2 | 0.00 | 1.00 | 1.00 | 3.00 |
| ROUGE-L | 2.00 | 2.00 | 7.00 | 11.0 |

The models chosen for testing were decided upon by their popularity, reputability, and applicability to the given tasks. In order shown above, the models used can be found on HuggingFace datasets as:

- google/mt5-base, facebook/mbart-large-50, Helsinki-NLP/opus-mt-en-de, facebook/wmt19-en-de

Unfortunately, despite the wmt19 translation model performing the best of all models tested, this model architecture is PyTorch based and is therefore out of the scope of this report’s course. Therefore, the Helinski-NLP model will be used for fine-tuning, due to both its TensorFlow architecture and high-performance capabilities displayed in the testing. The wmt19 model however, can be used for baseline performance comparisons as the best performing model. As all of these models achieve state-or-the-art results in their respective fields, and there is no model currently available for the task of cross-lingual summarization, the wmt19 model performance will be treated as the state-of-the-art performance representative for this section of the report.

**Fine-Tuning**:

In order to determine the number of epochs over which to train the model, the model will be fed a dataset of reduced size and otherwise base hyperparameters over many epochs, recording the ROUGE metric values at each epoch to find where over training begins to occur. This can be used as an indication of the ideal number of epochs to train the final model for.

The figure below shows the value of the Rouge metric recorded at each epoch. As is shown in the figure, the Rouge scores increase with the epoch, however, the greater the epoch the lesser the metric score average increase.

The testing shown above indicates that the Rouge metric scores begin to increase more slowly after 5 epochs. This shows that learning is decreasing, and this may be a suitable number of epochs over which to train. This value is also influenced by the limited computational capacity available.

The learning rate value was shown to slow down learning if too small and overtrain to the data if too large. After training for 5 epochs with a training rate of 0.00001 the average ROUGE-L score only just began to approach 14.2, while a training rate of 0.00003 shows a decline in Rouge metric scores on validation data after only 7-8 epochs. The learning rate when set to 0.00002 showed to be the most consistent, avoiding both overtraining and slower than necessary learning.

Similar to as with the first task; increasing the batch size over 8 results in memory issues, while decreasing it to 4 reduces model performance according by around 0.1 units per Rouge metric.

The decay rate of the model optimizer was set at 0.01 as no or little decay resulted in over training, while too great a decay rate prevented the model from learning past few epochs.

Given that this is a summarization task, an input length of 1024 was set to accommodate for the majority of the data input, and the output length was limited to 128 to accommodate for the majority of the summary data.

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric : Epoch** | ROUGE-1 | ROUGE-2 | ROUGE-L |
| 1 | 18.47 | 5.62 | 15.91 |
| 2 | 20.24 | 6.45 | 17.18 |
| 3 | 21.09 | 7.01 | 18.03 |
| 4 | 20.69 | 6.58 | 17.54 |
| 5 | 22.24 | 7.62 | 18.80 |

The table above shows the progression of ROUGE metric scores of validation data over epochs for the final training.

The newly trained model achieves a mean ROUGE-L F1 score of 18.6 over the testing dataset, with a standard deviation of around 5.9. In contrast, the best performing non-fine-tuned model; WMT19, achieved a mean ROUGE-L F1 score of 11.0, with a standard deviation of roughly 2.93. The t-test value was recorded at 3.36 with a corresponding p-value of < 0.004, meaning the result is significant at p < 0.05, providing evidence of statistical significance at the 95% confidence interval. It can therefore be concluded that this fine-tuned model performs better on cross-lingual summarization with an English source and German target language for the WikiHow dataset than the reference model WMT19.

**References**:

Lin, C. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. In *Meeting of the Association for Computational Linguistics* (pp. 74–81). <http://anthology.aclweb.org/W/W04/W04-1013.pdf>

Merritt, R. M. & NVIDIA. (2022). What Is a Transformer Model? *NVIDIA Blogs*. <https://blogs.nvidia.com/blog/2022/03/25/what-is-a-transformer-model/>

*Tokenizers - Hugging Face NLP Course*. (n.d.). <https://huggingface.co/learn/nlp-course/chapter2/4?fw=pt>