Game reviews abstractive summarization using Sequence to Sequence models

Julien MALKA
EPFL 3rd Year Student in Communication Systems
julien.malka@epfl.ch

Supervised by
Diego ANTOGNINI
EPFL PHD at LIA
diego.antognini@epfl.ch

Abstract

This Bachelor Thesis focuses on the task of abstractive multi-document summarization. We create a new dataset for this task with different properties than precedent datasets, and use known single-document summarization models on this dataset to see if they scale.

Introduction

Document summarization in computer science is the task that focuses on automatically finding important notions in a text and then presenting it in a human readable form while being factually consistent with the input, grammatically correct, and avoiding repetitions. Recently, some machine learning works have focused on ways to extractively or abstractively generate summaries of single-document inputs, that is respectively paraphrase the text or generate a new text, the latter being a much harder task. The task of multi-document summarization is similar to single-document summarization, but it takes as input a collection of documents instead of a single document. In this thesis, we create a dataset suitable for this task and in the continuity of the research from Liu et al. we experiment models from single-document summarization on this new dataset and evaluate their performance on this task. We try to evaluate the importance of the extraction phase on the final performance.

Related Work

Neural text summarization has been a big field of research these last few years. The difficulty of the task vary following the length of the input to summarize and the length of the summary to output.

Abstractive Text Summarization

One of the first works in the domain of text summarization was made by Rush, Chopra, and Weston (2015), in which they created a model to generate the headline of an article from its first sentences using the *English Gigaword* corpus (Graff and Cieri 2003). For this matter, they used an attention-based encoder to overcome the problem of losing information due to the length of the input.

Multi-Document Summarization

In the case of multi-document summarization, an interesting approach is to extract relevant pieces of information from the input documents, and the use some single-document summarization methods to produce a summary (Liu et al. 2018). Often, multi-document summarization datasets have much larger input length due to the fact that the task is to summarize multiple documents.

Game Dataset

Metacritics is a review aggregator website for video games. For a given game, the website gives access to a large number of reviews written by specialized journalists alongside with reviews written by users of the game. The site also provides a human written summary of the reviews. For a given game, the input to be summarized is then the collection of reviews written by the journalists, and the target summary is the summary provided by Metacritics.

Motivations to create a new dataset

This new dataset is very interesting because it allows us to see if the breakthrough discovered by Liu et al. scale well on an other dataset that has different properties. *Wikisum* is a corpus of more than 2 millions articles that are not taken from a specific domain, whereas our dataset is extracted from the gaming domain, and so comes with a specific vocabulary. The style of writing is also way more diverse on our dataset than on Wikipedia, and that may make the task harder. Finally, Liu et al. demonstrated that it was possible to generate Wikipedia pages with good performance when having the result of Google search for the title and the bibliography, but achieved poor performance when having only the result of Google search. This means that they can't automatically generate Wikipedia pages (because they would need a human to manually select the bibliography for the article). With our dataset we wish to automatically generate the lead of the Wikipedia page of a given game from the reviews of the game, we would then solve a real problem.

Technical solutions to build the dataset

Scraping

To create the dataset, we had to put in place a scraping pipeline. This pipeline needed to be efficient as there where more than a million reviews to download, analyse, convert to text.

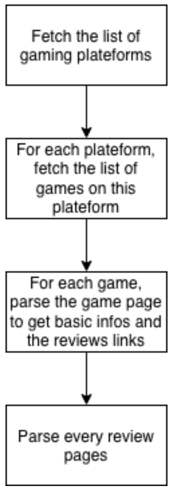

Figure 1: Pipeline used to scrape Metacritics

Extraction phase

For every game, we have a collection of text document from which we want to generate a summary. Just as human would do, we need to extract relevant information from these document before feeding the data to our model. For this matter we need an extraction algorithm. For this work, we used three different extraction algorithms and compared their performances.

- *cheating* : this extraction rank the sentences by highest Rouge-2 recall with reference summary. This extraction method helps us to evaluate the performances of the other extractors.
- *tf-idf* : We compute the *tf-idf* (Ramos 2003) with respect of the title of the game. For each word w_i of a document, we compute $N_{w,d} * log(N/N_{d,w})$ as a score for the word, where $N_{w,d}$ is the number of occurence of w_i in the document, N is the total number of documents in the corpus, and $N_{d,w}$ is the number of documents that contains w_i. We then summate the scores for the words in each sentence to get the sentence ranking.
- *custom extraction method* : Instead of using algorithmic methods to extact relevant information from the document corpus, we use state of the art models in extractive single-document summarization to get the most important sentences of each document before applying *tf-idf* to rank these sentences.

We then split the dataset in three parts : *train*, *validation* and *testing*. The sample from train are going to be fed to the model, those from validation are used to early stop the training, and those from testing are used to evaluate the performances of the model.

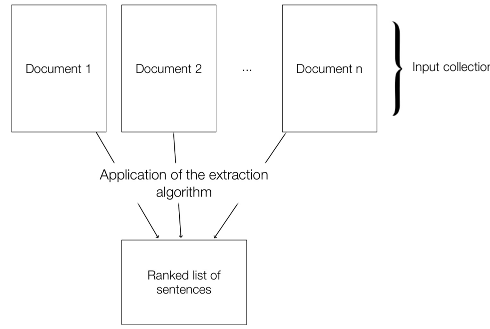

Figure 2: Principle of an extraction algorithm

Properties of the game reviews dataset

In this section we give a few properties of this dataset and how it compares to other dataset in abstractive summarization.

Dataset	Input	Output	Number of examples
Gigaword	10^1	10^1	10^6
Wikisum	10^2-10^6	10^6	10^6
Game reviews	10^6	10^2-10^3	10^4

Table 1: Order of magnitude of input length, output length, and number of examples of our dataset compared to past datasets

Our dataset has important input and output length while having relatively few samples to work with.

An other interesting property of our dataset that is not present in *Gigaword* and *Wikisum* is that our dataset do not have an universal style of writing like Wikipedia article have, hence it is way more difficult of a task to create an efficient model to summarize corpus of documents of this type.

Baseline model

Our baseline model is as simple random extractor from the vocabulary of the output. We randomly select a token from the output vocabulary until we selected enough tokens. The probability to select a given token is given by its appearance frequency. While this model will not generate grammatically correct sentences, or does not reflect the sense of the original text, it allows us to have a point of comparisons for the evaluation metrics that we are going to refer to later.

Sequence to Sequence model

We use a sequence to sequence neural network with attention decoder as described by Rush, Chopra, and Weston in the case of single document abstractive summarization. The model is a simple LSTM encoder/decoder architecture. We feed the tokenized input one by one through the encoder, giving us a sequence of hidden states h_i. For each token to decode, we then feed the weighted sum of the h_i's to the decoder, where the weights comes from the attention network, and get a probability distribution over the output vocabulary for each token of the sequence. For complexity reasons, we run the model with limited input length : 500 tokens and 1000 tokens.

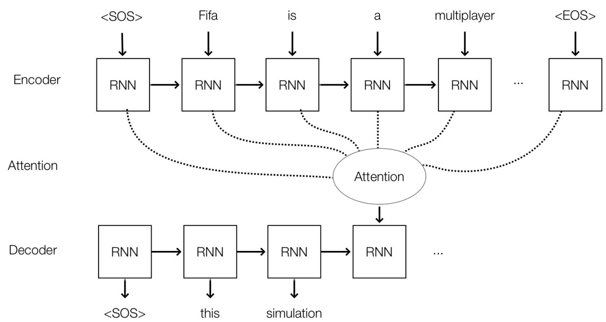

Figure 3: Sequence to sequence with attention model

Summary reconstruction

When the model is trained, we use a beam search algorithm (Wu et al. 2016) to generate the summaries.

Input: k the depth of the beam search, The trained model
Result: A summary
candidates = [];
foreach *token to decode* **do**
  foreach *candidate in candidates* **do**
    get the probability distribution for the next token from the decoder;
    take the k best token and add them at the end of the candidate, yielding k candidates;
    the score for each new candidate is the score of the candidate multiplied by the probability of the token
  end
end
return *the candidate with the higher score*

Algorithm 1: Beam Search

To test the efficiency of the beam search and its involvement in the performances, we will generate the summaries with beam search (we use $k = 5$) and with a greedy decoder (equivalent to $k = 1$).

Results

Rouge metric

After training the models, we generate summaries for games of the testing part of our dataset. We then use the Rouge metric (Lin 2004) to evaluate our summaries.

Extraction	NLL Loss	Rouge-1	Rouge-L
cheating	1827.68	40/13/18	39/12/18
tf-idf	1463.64	13/11/12	13/11/11
custom	/	/	/

Table 2: Rouge metrics for the summaries when the input and output were limited to 500 tokens

Extraction	NLL Loss	Rouge-1	Rouge-L
cheating	1827.68	31/24/27	30/23/25
tf-idf	1463.64	13/13/13	12/12/12
custom	/	/	/

Table 3: Rouge metrics for the summaries when the input and output were limited to 1000 tokens

Unfortunately, I was only able to train a model to generate the ranking with the custom extractor, but the timing was too short to preprocess the dataset again and train the sequence to sequence model on it, and we have no rouge score for this extractor.

We can see that the Rouge scores when using *tf-idf* are way inferior that scores with the *cheating* method. This means that there is a large space for improvement in the extraction phase. This is the reason it could be interesting to have the results for the *custom* extractor.

k used for the beam search	Rouge-1	Rouge-L
1	19.66	18.25
5	25.54	23.48

Table 4: Rouge metrics for the summaries in function of the k used for the beam search (200 tokens)

We can see that the use of the beam search algorithm to generate the summaries increases the Rouge performance of our model. It can also be noted that for k smaller than 5, the increase in performances was not notable.

Model	Rouge-1	Rouge-L
Seq2Seq-attention	25.54	23.48
Baseline Model	25.25	22.38

Table 5: Rouge metrics in function of the model used

This results shows us that while Rouge is an interesting metric to evaluate generated summaries, is should not be taken as the only metric, as summaries generated by our Baseline Model get a similar Rouge score than our model, but of course are not good summaries at all. Human evaluations are also very important.

Generated summaries

Reference Summary : what is final fantasy xv : pocket edition ? it seems like a pretty simple question , but figuring out the answer is actually rather difficult . you could say it was a dem ke , but since it only runs on high - end devices , which cost more to buy as standalone products than the consoles the game was first released on , that doesn ' t actually make much sense when you think about it . so does that make it a standalone experience ? well , no , because it ' s still the same story , albeit told with fewer polygons and a less interactive battle system . and above and beyond that central query , there ' s another question we need to answer here . is final fantasy xv : pocket edition worth buying ? to be honest , that ' s a toug ie as well . the good prince okay , let ' s get the good out of the way first . the game is surprisingly charming , and its art - style makes it feel more like a ff game than the flashy polygons and endless vistas of the original . it ' s fun as well , and rather than trying to shoehorn the console experience onto your phone or tablet , it tai ors things for the devices it ' s been designed for . so where in the original you ' re controlling noctis with a variety of buttons pushes , joystick presses , and trigger pulls , here a lot of that is taken out of your hands . tap and you ' ll attack a monster automatically , leaving you to concentrate on guard breaks , special dash moves , and using the super - moves of your cohorts . that ' s not to say that the battles are spectator affairs . there ' s still a lot for you to do while your party of royal dude - bros smashes monsters with swords . you can swap weapons , heal , change targets , and all manner of other things . and it works really well . it takes the basis of the ffxv system and turns it into something that actually feels good on a touchscreen . and judging by some of square enix ' s other attempts at mobile ports , that ' s no mean feat . when you ' re not fighting you ' re navigating the world with taps . poke a finger on the screen and noctis will find a route to get there . it means things like getting stuck on scenery tend not to happen , which is nice .

Generated Summary : final fantasy xv is an open world action role - playing game set in an open world environment and played from a third - person perspective . the noctis luc s ca lum protagonist to gladiolus , the noctis , the the , the the the the the the the the the the the the the the the the the the the the the the the the the the the the the [...]

Reference Summary : evil dead : regeneration. the introduction of sam (voiced by ted ra mi) as a half - human , half - dead partner for ash is actually for the benefit of the game, as it allows for heavy doses of comic relief as well as some twists in the gameplay . sam can be used in all sorts of ways including being kicked onto adversaries (where he ' ll rip their heads off) and he can also fend for himself with basic melee and projectile attacks . he ' s also helpful in certain situations in which ash will possess him and be able to control his movements ; this accommodat s access to certain level areas that are out of reach for ash , but not for the three - foot - high sam . usually , these sequences involve getting sam to an area where he can pull a lever to gain ash access as well , but sometimes sam can drop onto large mini - boss characters in order to ride them around , thus clearing obstacles and eliminating the mini - boss . sam is really quite an inspired addition to the game , as his and ash ' s banter is quite funny , with lots of in - jokes about the characters , the actors who play them , and the game itself . you are also heavily encouraged to kill sam in various ways , due to the fact that the he regenerates , thanks to his half - dead te form . this allows for many comedic moments of kicking sam into griners , fans , spikes , and so on . often , sam will have accomplished a task and you will see him ? in a cutscene ? meet his maker in comical fashion . once again , though , he will reappear right next to you and can be used over and over . this mechanic is actually quite helpful for the game , as sam can constantly retry certain sequences without ? dying ? and forcing you to reload a save or redo tedious gameplay . fighting dead tes will take up most of your time in this game , but there are some simple platforming sections to mix things up . unfortunately , these sections don ' t play well at all , since ash ' s jumps are stiff and awkward .

Generated Summary : evil dead is an action - adventure action game . the player - the the the the the the the the the the the the the the the the the the the the the the , the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the [...]

The generated summaries are relevant only for one or two sentences, that are often on point, and grammatically correct. After, the model outputs becomes irrelevant. It could be due to the very different styles of writing, to a weak model, or to the difficulty of the task on this dataset.

References

Graff, D., and Cieri, C. 2003. English Gigaword.

Lin, C.-Y. 2004. Rouge: a package for automatic evaluation of summaries.

Liu, P. J.; Saleh, M.; Pot, E.; Goodrich, B.; Sepassi, R.; Kaiser, L.; and Shazeer, N. 2018. Generating Wikipedia by Summarizing Long Sequences. *arXiv e-prints* arXiv:1801.10198.

Ramos, J. E. 2003. Using tf-idf to determine word relevance in document queries.

Rush, A. M.; Chopra, S.; and Weston, J. 2015. A Neural Attention Model for Abstractive Sentence Summarization. *arXiv e-prints* arXiv:1509.00685.

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; Klingner, J.; Shah, A.; Johnson, M.; Liu, X.; Kaiser, Ł.; Gouws, S.; Kato, Y.; Kudo, T.; Kazawa, H.; Stevens, K.; Kurian, G.; Patil, N.; Wang, W.; Young, C.; Smith, J.; Riesa, J.; Rudnick, A.; Vinyals, O.; Corrado, G.; Hughes, M.; and Dean, J. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv e-prints* arXiv:1609.08144.