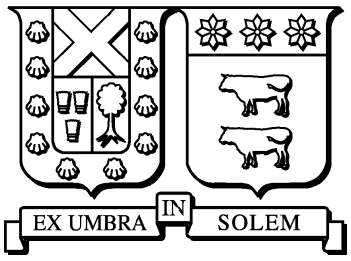
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE INFORMÁTICA

SANTIAGO – CHILE



"NEURAL ABSTRACTIVE SUMMARIZATION OF ONLINE NEWS DISCUSSION THREADS"

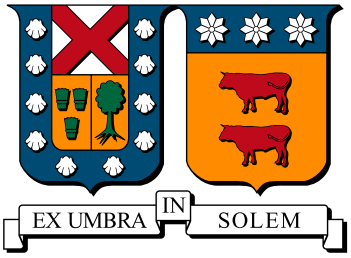
IGNACIO EDUARDO TAMPE PALMA

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL INFORMÁTICO

PROFESOR GUÍA: MARCELO MENDOZA

NOVIEMBRE 2020

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO – CHILE



"NEURAL ABSTRACTIVE SUMMARIZATION OF ONLINE NEWS DISCUSSION THREADS"

IGNACIO EDUARDO TAMPE PALMA

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL INFORMÁTICO

PROFESOR GUÍA: MARCELO MENDOZA

PROFESOR CORREFERENTE: EVANGELOS MILIOS

NOVIEMBRE 2020

Acknowledgements

I would like to thank my family. They are a core component of my life. I wouldn't be here if it wasn't for their constant support and inspiration, allowing me to become the professional I have always aspired to be. Thank you, mom; I will always admire your tenacity to have raised 2 kids by yourself while also having to work. You taught me to always strive for greatness and fight for what is just.

Thanks to my friends, we have come a long way since we started university up until now. You made it easier to study for a test on a challenging course. When times were stressful, you always provided a reassuring: "I haven't started working on that assignment either". Plus, the parties and beers we shared made everything more fun. I wish you all the best in the times to come!

I would also like to acknowledge professors Mendoza and Milios, they guided me during this process and pushed me to go further on my work, the ELAP scholarship allowed me to work on the MALNIS lab to develop my thesis and have access to the ComputeCanada Cluster which was instrumental to train my models.

Finally, a special thanks to Felipe González, my boyfriend. He has always been there to support me when times were down and has been the first to hug me when it was time for celebration, you are someone I look up to a lot, and I know the future is bright ahead of us. I love you!

Dedication

To Felipe
Soulmates are a special thing
And I know it when I see you by my side.

And to my late dog Oscar
Growing up with you was the best
I wish I could have celebrated
this moment with you too.

Resumen

Resumir automáticamente normalmente ha dependido de resúmenes realizados por expertos para poder entrenar modelos. Las redes sociales agregan nuevos desafíos a las técnicas de resumen ya que requieren revisar enfoques multi-documento y multi-autor. En este trabajo revisamos esta tarea introduciendo un método que genera resúmenes abstractivos de discusiones en artículos de noticias en línea. El método extiende a una arquitectura basada en BERT, incluyendo una capa de atención a la que se le entregan los "Me gusta" de los comentarios durante entrenamiento. Para entrenar el modelo, se definió una tarea que consiste en la reconstrucción de comentarios de alto impacto, basados en su popularidad. De esta forma, el modelo logra aprender a resumir discusiones basado en los comentarios más relevantes. Nuestro novedoso enfoque provee un resumen que representa los aspectos más importantes de la noticia que la gente comentó, incorporporando el contexto social como base de información para hacerlo. Nuestro modelo fue validado con ROUGE Recall entre el resumen generado y cada uno de los comentarios del hilo, ponderados según popularidad. Basada en esta evaluación, nuestro modelo con su capa de atención, sobrepasa significativamente a modelos base extractivos y abstractivos sin atención.

Abstract

Summarization has usually relied on gold standard summaries to train extractive or abstractive models. Social media brings a hurdle to summarization techniques since it requires addressing a multi-document multi-author approach. We address this challenging task introducing a novel method that generates abstractive summaries of online news discussions. Our method extends a BERT-based architecture, including an attention encoding that fed comments' likes during the training stage. To train our model, we define a task which consists of reconstructing high impact comments, based on popularity (likes). Accordingly, our model learns to summarize online discussions based on their most relevant comments. Our novel approach provides a summary that represents the most relevant aspects of a news item that users comment on, incorporating the social context as a source of information to summarize texts in online social networks. Our model is evaluated using ROUGE recall between the generated summary and each comment on the thread, weighted by its popularity. Based on such evaluation, our model, including the social attention encoding, significantly outperforms both an extractive summarization baseline and a baseline abstractive model without an attention encoding.

Content Index

List of Tables

List of Figures

Glossary

- **NATS**: Neural Abstractive Text Summarization
- **LM**: Language Model
- **OOV**: Out of Vocabulary
- **XENT**: Crossentropy
- **Seq2Seq**: Sequence-to-sequence
- **LCS**: Longest Common Subsequence
- **TF-IDF**: Term Frequency–Inverse Document Frequency
- **BERT**: Bidirectional Encoder Representations from Transformers
- **ROUGE**: Recall-Oriented Understudy for Gisting Evaluation
- **LD**: Lexical Diversity
- **STD**: Standard Deviation

Introduction

Social media has become a major source of information and opinions worldwide. Per-minute, Internet users post over 500,000 tweets, more than 188,000,000 emails and 18,000,000 texts. The world's internet population has reached 4.39 billion people, over half of the world's population[1]. Digesting and analyzing this ever-growing amount of textual information can become very difficult for humans to comprehensively analyze, as corpora can easily consist of gigabytes of raw data. Given this situation, automated mechanisms to extract meaningful information become relevant to handle these text datasets.

The automatic construction of summaries from texts retrieved from web forums, online news websites, or blogs encompasses many challenges. Unlike formal documents (e.g., scientific papers), discussions and online conversational threads tend to be informal, using and abusing slang expressions, special symbols (e.g., hashtags and/or emojis), and other types of language specificities that characterize web communication. [1] show that these texts present greater challenges to automatic summarization techniques, since as conversation progresses, posts may change, exhibiting topic drifts and topic dependencies from previous opinions. Also, as posts tend to be short, classical text representations (e.g., Tf-Idf) can be tackled by data sparseness [3].

The use of automatic techniques for constructing text summaries has had a wide development in the last years. These techniques are based on two main approaches: sentence extraction [7] or abstraction of a new text that condenses the original document's content [14]. The rise of deep learning has pushed the development of more accurate language models that

[1]Data Never Sleeps 7.0, Domo Inc. Accessed on 09-03-2020, `https://www.domo.com/learn/data-never-sleeps-7`

help implement abstractive summarization techniques [22]. The last years have shown that abstractive techniques based mainly on the seq2seq [20] architecture have reached state of the art results in these tasks [23].

Despite the success of neural abstractive summarization techniques, its impact in summarizing web discussions is less explored. This gap is because many of the abstractive methods follow a single document approach, considering the comments as a complementary source of information to improve the seminal document description [6]. Accordingly, these approaches discard misalignment between the discussion and the original subject, focusing the analysis and the validation in improving the seminal description of the topic. Also, architectures based on self-attention mechanisms [21] appear in social context summarization scenarios as less explored.

This work focuses on studying the possibilities that the Transformer architecture [21] has to summarize news articles' discussions. To do this, we combine BERT [4], a pretrained language model text encoder, with a Transformer-based decoder. We extend this architecture, including an attention encoding that is fed with users' likes. The attention encoding gets the focus of the model into the comments with the highest social impact. Accordingly, the model summarizes the most relevant aspects of a discussion gathered by a news article. This novel approach differs from its closest competitors because a great part of the prior work focuses on identifying salient comments that contribute to a better understanding of the original news item. Our approach has a different motivation. We address a multi-document multi-author approach to summarize the discussion. Then, instead of discarding the potential misalignments between the news and the user's comments, we pay attention to salient comments favouring identifying new aspects related to the news. Accordingly, in our approach, the users are considered co-authors of the original news, to which they can add new information. To identify salient comments, we pay attention to users' likes. We claim that high impact comments reveal which comments are most relevant to describe and extend the original news content. We define a text summarization task that helps predict high impact comments, providing the news title as part of the target text. To encourage the model to pay attention to salient comments, we introduce a data-driven strategy that focuses the model into these comments. We evaluate our model using ROUGE scores [13] on a new dataset

developed by us that provides news descriptions, comments, and likes. Based on such evaluation, our model, including the social attention encoding, significantly outperforms both an extractive summarization method and a neural abstractive model.

The specific contributions of our work are:

- We extend an encoder-decoder approach based on the Transformer Architecture, including an attention encoding that pays attention to salient comments.
- We introduce a novel approach for text summarization. The summarization task's target includes detecting salient comments, extending prior single document approaches to a multi-document multi-author text summarization scenario.
- We release a new dataset that includes news descriptions, reader comments, and likes, favouring reproducibility, and further extensions to our proposal.

The document is organized as follows. In chapter 2, we discuss related work in text summarization with a specific focus on social context summarization. In Chapter 3, we introduce our model and the experimental methodology. Experiments are reported in Chapter 4. We conclude in Chapter 5, providing concluding remarks and finally, In chapter 6 we discuss future work.

Chapter 1

Problem Definition

Social media has become a major source of information and opinions worldwide. Per minute, Internet users post over 500,000 tweets, more than 188,000,000 emails and 18,000,000 texts. The world's internet population has reached 4.39 billion people, over half of the world's population.[1] And forecasts expect that these numbers will only keep on increasing.

Digesting and analyzing this ever-growing amount of textual information can become very difficult for humans to comprehensively analyze, as corpora can easily consist of gigabytes of raw data. Given this situation, automated mechanisms to extract meaningful information become relevant to handle these text datasets.

1.1. Automatic Text Summarization

Automatic Text Summarization refers to a technique for shortening a long text corpus without human intervention. The idea is to create a coherent and fluent summary which both preserves the important information and meaning of the documents. Having this summaries allows for the content to be retrieved, processed and digested efficiently by further analytical tools or people. The capability of Artificial Neural Networks has shifted most of the current

[1] Data Never Sleeps 7.0, Domo Inc. Accessed on 09-03-2020, https://www.domo.com/learn/data-never-sleeps-7

work into applying *Deep Learning* Techniques which has greatly improved readability and salience of the generated summaries [18].

There are two main strategies to tackle the automatic summarization problem: *Extractive* and *Abstractive*. A method is called *extractive* if the output sentences are selected directly from the source corpus. Usually, this kind of solutions deal with a binary classification problem: whether the input sentence is relevant or not, similar to highlighting phrases, this method requires an annotated corpus for training. In the case of *abstractive* summarization solutions, the generated summary is a novel text that could be completely different from the input as it can include new words and sentences generated by language generation models. They have a bigger potential of producing a higher quality summary because the language model can learn how to correctly phrase and link information, which improves readability and relevance, as the task is to imitate human-written summaries as ground truth.

1.1.1. Abstractive Text Summarization

On the recent years, advancements on *Deep Learning* has brought new possibilities that enabled a surge of Abstractive Text Summarization methods. The rise of sequence-to-sequence (*seq2seq*) neural network models and convolutional neural networks (CNN) have the ability to process entire sentences by including memory modules that allows them to not loose track of the input document, expanding the length of data that can be processed. providing a better context for the model to understand the input corpus. Aside from their advantages, these models have to deal with a lot of challenges that arise from abstractive methods: (1) handling Out Of Vocabulary (OOV) words, the set are unknown by the language model and therefore the network cannot deal with topics outside the ones provided for training; (2) they cannot reproduce salient information accurately because the attention mechanism also fail with OOV words (3) the output summaries usually suffer from repetitions of word or phrases, resulting in unnatural writing. Attempts to solve the first two problems have been made by referencing words from the source text to the output, producing factual information with novel words unknown to the network. The third problem has been addressed by a coverage mechanisms and heuristics that penalize repetition and ensure a broader coverage of

the input document.

1.1.2. Summarization on Social Media

The expansion of online social platforms brings in consequence a explosion of new applications, nowadays users review products, places and services, they discuss news articles and talk on on dedicated forums like Reddit. In both the perspective of an user and a company, reading through all the reviews and comments can become a never ending task and if you only read a few of them, you might become biased. Plus, discussion sites are important for search engines for information retrieval tasks, the problem is that it will give the user an entire thread of information which they will most likely read just a small portion of it, skimming over it. As the thread's length increases, the need for tool-assisted summarization becomes more and more relevant [1].

Online discussion is a novel setting to apply abstractive summarization that brings it's own challenges. First, there is no clearly defined summary to use as ground truth. Also, information in discussion threads can be scattered randomly along the thread. This differs from news articles where the distribution of information is heavily biased towards the first paragraph [9].

1.1.3. Motivation

The summarization of online news discussions poses several challenges. Users' comments are generally in colloquial language. Accordingly, the extraction of relevant information from these messages requires an additional effort. Also, many times, user comments can be uninformative. Many times, the news generates a large flow of comments, many of them repetitive. A summarizer who considers the social context must grapple with these difficulties.

We claim that relevant information receives likes and that less informative comments attract less attention from users. This statement is correct if all messages have the same chance of being observed. In the case of digital news media, many comments are published on the

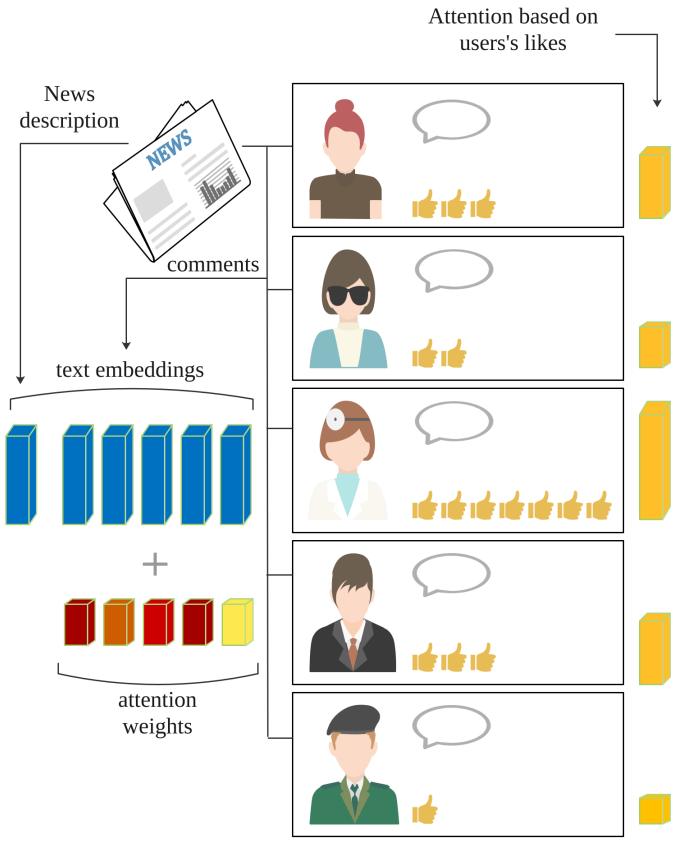


Figure 1.1: In our model, the attention weights are driven by the user's likes. The description of the news and the user's comments are embedded in the architecture using text embeddings. The architecture includes an attention encoding that enhances salient comments.

same news site, and therefore, users can read these comments simultaneously as they review the descriptions of the news. With low editorial mediation, user comments can attract as much attention from the community as the original story in this type of site. In this scenario, we claim that likes reveal post informativeness. Accordingly, we propose to use the likes to drive the attention of an automatic summarization method to the most salient posts (see Figure 1.1).

The key element of our proposal is the definition of a like-based attention encoding. We use a summarization architecture fed with news' descriptions (e.g., news' titles and subheads), and users' comments. Unlike related work, in which user comments are selected in terms of

their relevance to the news, we pay attention to the community's reaction. Figure 1.1 shows that comments with many likes will generate a greater attention weight than those comments with few likes. The attention based on users' likes will provide more information to the encoder, highlighting these comments during the encoding process. As these comments will be more relevant when coding the news and its comments, the decoder will reconstruct a social context summary giving more relevance to the comments with the most likes.

Chapter 2

Related Work

2.1. Automatic Abstractive Summarization

Shi et al. [18] give a good overview of the abstractive summarization method, including it's challenges and current solutions to overcome them. Most notably, they mention how *Pointer-generator networks* allow a model to reference words from the input sequence even if they are not part of the vocabulary, this helps to solve the problem of OOV words and makes the model more robust for unseen information. Aside from *seq2seq* models they include Convolutional architectures, decoding methods like beam search and different training schemas by including reinforcement learning and multi-tasking.

Subramanian et al. [19] present a novel approach for abstractive summarization of long scientific documents by discarding *seq2seq* models and using a transformer instead, this proved to outperform traditional copy mechanisms achieving higher ROUGE [13] scores. To achieve this state-of-the-art performance, by doing a two step process, first using a extractive method to select relevant sentences and then summarize the chosen sentences. Aside from ROUGE scores, they demonstrate the abstractiveness of their generated summaries is higher than those from copy mechanisms which tend to rely too much on copying large chunks of information from the source. The downside to these results is the extreme computing power and time required to train these models, they spent 5 days on 16 NVIDIA V100 GPUs

training a language model.

Using pretrained language models, like the work on [15], dramatically reduces the computing power needed to solve this task, on this work, they propose a way to employ a pretrained BERT model as an encoder, which can be fine-tuned for the task, and a transformer-decoder from [21] to be trained for the task. They also propose a fine-tuning scheduling technique using different optimizers for both encoder and decoder in order to get state-of-the-art results.

2.1.1. Validation

Metrics for generated summaries need to be able to evaluate both topic coverage and factual truth. As of today Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [13] is considered to be the gold standard metric for text summarization. ROUGE focuses on overlapping n-grams, word sequences and word pairs between the generated summary and the ground truth summary created by a human. There are four different ROUGE measures:

- ROUGE-N: consists of N-gram co-occurrence statistics between the tested and original summaries, the value is obtained by:

$$\frac{\sum_{S \in \{Reference\ Summaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{Reference\ Summaries\}} \sum_{gram_n \in S} Count(gram_n)} \tag{2.1}$$

  Where n stands for the length of the n-gram, $gram_n$, and $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and a set of reference summaries. As it's based on co-occurrence, it measures a recall score of the generated summary.

- ROUGE-L: F-Score based on the Longest Common Subsequence (LCS) between sentences from two summaries. Let X be a sentence of the generated summary and Y a sentence of the origial summary of length n and m respectively, the F_{lcs} score is

calculated as:

$$R_{lcs} = \frac{LCS\ (X, Y)}{m} \tag{2.2}$$

$$P_{lcs} = \frac{LCS\ (X, Y)}{n} \tag{2.3}$$

$$F_{lcs} = \frac{\left(1 + \beta^2\right) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \tag{2.4}$$

Where $\beta = P_{lcs}/R_{lcs}$. By using LCS, you can extract in-sequence matches that reflects word order in a sentence level and does not require a n-gram length to be predefined. This characteristics allows for sentence level structure to be captured.

To obtain a summary-level LCS score, ROUGE-L takes the union of the sentence-level LCS matches of a candidate summary sentence and every sentence from the ground truth summary.

$$R_{lcs} = \frac{\sum_{i=1}^{u} LCS_{\cup}\left(r_i, C\right)}{m} \tag{2.5}$$

$$P_{lcs} = \frac{\sum_{i=1}^{u} LCS_{\cup}\left(r_i, C\right)}{n} \tag{2.6}$$

$$F_{lcs} = \frac{\left(1 + \beta^2\right) R_{lcs} P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \tag{2.7}$$

Here $LCS_{\cup}(r_i, C)$ is the LCS score of the union longest common subsequence between reference sentence r_i and candidate summary C.

- ROUGE-W: Weighted Longest Common Subsequence, (WLCS) This metric takes the implementation of ROUGE-L but considers consecutive matches as a higher weight. It's computed by a dynamic programming table, which will output the result of the WLCS algorithm. A weighting function f is applied that must comply with $f(x+y) \geq f(x) + f(y)$ which results in consecutive matches receiving higher scores. Considering

this, the F-score is obtained as follows:

$$R_{wlcs} = f^{-1}\left(\frac{LCS\ (X,Y)}{f\,(m)}\right) \tag{2.8}$$

$$P_{wlcs} = f^{-1}\left(\frac{LCS\ (X,Y)}{f\,(n)}\right) \tag{2.9}$$

$$F_{wlcs} = \frac{\left(1+\beta^2\right)R_{wlcs}P_{wlcs}}{R_{wlcs}+\beta^2 P_{wlcs}} \tag{2.10}$$

- ROUGE-S: Skip-Bigram Co-Occurrence Statistics, this metric matches the occurrence of bigrams that are not written consecutively in the sentence (allowing for skips), similarly to previous ROUGE scores, it's computed by:

$$C(n,r) = \frac{n!}{r!\,(n-r)!} \tag{2.11}$$

$$R_{skip2} = \frac{SKIP2\,(X,Y)}{C\,(m,2)} \tag{2.12}$$

$$P_{skip2} = \frac{SKIP2\,(X,Y)}{C\,(n,2)} \tag{2.13}$$

$$F_{skip2} = \frac{\left(1+\beta^2\right)R_{skip2}P_{skip2}}{R_{skip2}+\beta^2 P_{skip2}} \tag{2.14}$$

  Where $SKIP2(X,Y)$ is the number of skip-bigram matches between X and Y, β controlling the relative importance of P_{skip2} and R_{skip2}, and C is the combination function. Limits for the skip length should be considered because it could loose any semantic relevance to study bigrams that are too far apart on longer documents. Extensions for different n-gram lengths are included.

Extensions

Even though ROUGE is considered the standard to evaluate generated summaries, it's bias to rely on surface lexical similarities causes problems when scoring the results of abstractive or paraphrasing methods, as polysemy and synonymy are not addressed at all on these metrics, because context is not considered when dealing with words, as of today, ROUGE is over 15 years old and several advancements on Natural Language Processing has been made. The inclusion of word embeddings as vector representation of words has expanded the Language Model capabilities.

On [16], the authors propose ROUGE-WE, an extension to the original ROUGE metrics that include a word embedding vector space to compute semantic similarities between words. Considering f_R as the function that matches words in ROGUE, the score would be defined as:

$$f_R(w_1, w_2) = \begin{cases} 1, & \text{if } w_1 = w_2 \\ 0, & \text{otherwise} \end{cases}$$

This binary output results in synonyms not being recognized and therefore considered as two completely different words. For example, the following sentences: *dogs are pretty* and *puppies are beautiful*, refer to the same idea semantically but word-wise are completely different with no skip-bigram co-occurrence. The inclusion of word embeddings takes the vector representation of words $w \rightarrow \vec{v} \in \mathbb{R}^n$ and compute the similarity between words using dot product $\vec{v}_1 \cdot \vec{v}_2$, using this we can update f_R to obtain $f_{WE} \rightarrow \mathbb{R}$:

$$f_{WE}(w_1, w_2) = \begin{cases} 0, & \text{if } w_1 \text{ or } w_2 \text{ are OOV} \\ \vec{v}_1 \cdot \vec{v}_2, & \text{otherwise} \end{cases}$$

The effectiveness of this output is not without bias though, the vector spaces are trained on data and therefore are subjected to its content. Changing the underlying embedding will inevitably result in a different score from the function which makes it intrinsically not deterministic.

2.2. Summarization in Social Contexts

Kim, et al. [9] apply abstractive summarization on short stories from reddit, for this task, they propose an architecture referred to as *multi-level memory* network that extracts in between representations of words, sentences and paragraphs to have a hierarchical representation of the document. They also do an important analysis of the distribution of gold summary words on the input sequence on their dataset and the common summarization datasets, they show how there is a bias towards the first paragraph in news articles. This is further confirmed by Kryscinski, et al. [11] by proving with human annotators how this bias can be *exploited* to

get better metrics. ROUGE [13] is also criticized because it cannot ensure factual information or deal with synonyms, ROUGE-WE can attempt to solve that later one by using word embeddings (WE).

It's important to understand the differences between summarizing structured news articles or scientific papers in comparison to online posts, Almahy et al. [1] focus on properly characterize the structure of the online post and how it differs from a news article, considering the asynchronous communication between participants, formality of the conversation and hard to identify sentence boundaries. They include the challenges that come with dealing with forum discussions such as:

- Topic Dependencies: Posts are not independent of each other, they could be a reply to another previous one, the thread is not just a linear sequence of text but a forest of replies.
- Topic Drifting: As the conversation progresses, subtopics emerge and expand on the original idea, but could potentially lead to a whole different theme.
- Text Sparseness: Posts can be short in length, which makes co-occurrence harder to rely on for text representation methods like TF-IDF.

Finally they discuss on summarization techniques, mainly using graph-based solutions, although they not include *machine learning* methods in depth.

While most of the work is focused towards summarizing facts, Ying et al. [5] attempt to extract the general opinion from forums corpora, because of this, they propose several heuristics to filter out bad quality posts in order to prioritize well-written posts for summarization, these criteria include sentence features like:

- Representativeness over the thread.
- Text quality based on average word and sentence length, usage of punctuation marks and emoticons, POS tagging and parse tree depth.
- Subjectivity by counting sentiment words in a sentence using a lexicon.

By using metrics they can properly select the best posts on the thread for the summarization task. To generate the summaries, they applied a global optimization model by integer linear programming which results in extractive summarization by maximizing concept coverage.

Some summarization techniques applied to social media focus on solving single document problems. For instance, [14] uses a convolutional neural network to include a text's global encoding, providing context to a seq2seq architecture. The authors show good results summarizing short posts collected from Sina Weibo. [22] also uses the convolutional seq2seq architecture for text summarization, showing its success in many benchmark data. [9] propose an abstractive summarization method of short stories from Reddit. Their architecture, referred to as *multi-level memory*, extracts representations of words, sentences, and paragraphs producing a hierarchical encoding of each story. The proposal gets good results, but the authors show that many of the summaries are biased towards each story's first paragraphs. This characteristic matches many of the gold standard summaries, which consider each document's first sentences a good outcome. This finding coincides with the results found by [11], whose proving with human annotators how this bias can be exploited to get better metrics. These findings have pushed the focus of the problem towards the criticism of how the quality of an automatic summarization method is evaluated. For example, ROUGE [13] has been criticized because it cannot distinguish between factual and subjective information, being also tackled by homographs [18].

[5] summarize opinions from web forums. To fulfill this purpose, they introduce some features to filter out low-quality posts. These features consider lexical coverage, stylographic-text features, and subjectivity measured in terms of sentiment words. Using linear programming, they combine these features to implement an extractive summarization method. The results show that the three criteria are useful for an extractive method of posts in web forums.

[6] examine how to include reader comments to improve news articles descriptions. The authors study the distribution of information on comments incorporating these data into the summarization process. They introduce a seq2seq-based model named *reader-aware summary generator* (RASG), which considers an attention layer that fed a semantic alignment score between the reader comments words and document words. The type of training used by RASG is based on adversarial learning. Experimental results show the feasibility of RASG

improving the description of the news articles.

Recently, [12] introduced an abstractive summarization model to describe social events on Twitter. The proposal uses a pretrained BERT encoder to code the tweets. Also, they fed an ”Event Embedding” that captures the whole data stream. All this information is fed to a transformer-based decoder that generates the summary. The authors define a supervised task, grounded on human annotators that created a dataset of gold summaries for each event. Experimental results show that the proposal is feasible.

Connection with the related work. Many text summarization methods are grounded on the seq2seq architecture, and those approaches are based mainly on a single document single author scenario. When the social context is considered, some approaches pay attention to the description of an original subject, looking for alignments between the reader's comments and the original document. These models are based on single document approaches. Related work shows fewer efforts focused on identifying salient comments. [5] use an extractive multi-document approach to identify these comments. [6] and [12] propose abstractive approaches to address this problem but in single document contexts. Specifically, [6] takes the reader's comments in a seq2seq architecture but discards the comments and the original document's misalignments. [12] is probably the closest approach to our proposal, as it uses the Transformer architecture to build event summaries on Twitter. However, it does not use users's likes to identify salient comments. Instead, the model is trained to generate human experts' target texts, focusing the model in generating a single document output that summarizes the whole event.

Chapter 3

Methodology

3.1. Data

The data consists of spanish comments obtained directly from *Emol*[1], the online version of the Chilean newspaper "*El Mercurio*" to study our model. The news media provides open data along with comments from its users. Also, users can vote for the comments of other users through the mechanism of likes. The data is openly published by the media using JSON format. The data was downloaded directly from the site. To comply with data privacy protocols, user identities were anonymized.

Users can create an account on the site or associate a Facebook or Twitter account within the website. All users with publishing rights do so with authentication. There are no anonymous comments. The website provides its users with mechanisms to comment on each news. Users can comment directly on a story. The site keeps a record of the comments related to each news item published by the digital media. The website provides a voting mechanism for comments, which considers likes. Site users widely use this mechanism. Other interaction mechanisms are provided by the website, such as blocking users and reporting comments, which have less use. In addition to commenting on news, the users can keep a wall. In this place, opinions of general interest can be published. Users can follow other users to access

[1] `https://www.emol.com/`

their walls. However, the social graph associated with this interaction mechanism has low density, so it is of little interest to our study.

The entire dataset comprises 143340 news retrieved from April 1, 2016, to April 20, 2019. Of this set, 122778 news has comments. The news dataset contains different categories: National, International, Technology, Economy, Entertainment. We focused our work in the national news category, consisting of 41733 articles. This subset was partitioned into three folds: one for training with 35024 examples, one for validation consisting of 3354 news articles and 3355 articles for testing. The dataset partition was done randomly.

3.1.1. Text Preprocessing

The comments were cleaned using Python and *regex* to remove any HTML tags and symbols. We remove people's mentions, URLs, laughter, and repeating punctuation. After this, comments with at least five words were included. We did not remove stopwords as they help with readability in the summary generation. The x news included in the training partition records x comments. The x news included in the testing partition records x comments.

Token sequences were generated using Standford CoreNLP4. Finally, each word from the input was processed using the BERT subword tokenizer. This tokenizer splits words into subtokens from its pretrained vocabulary, mitigating the out of vocabulary problem. We worked with sequences of 512 subtokens, as is usual in BERT [4].

3.2. Architecture

To encode data in Spanish we used BETO [2], a BERT model trained on a Big Spanish Corpus[2], available from the HuggingFace Transformers Library [3]. Let x_0 be the news description provided by a digital media website, and let $x_1 \dots x_n$ be the reader's comments. To feed these texts into a Transformer-based encoder, we build a text sequence. We organize a

[2] https://github.com/josecannete/spanish-corpora
[3] https://huggingface.co/dccuchile/bert-base-spanish-wwm-uncased

token sequence to keep the original structure of comments safe, providing a matrix of tokens with $n + 1$ rows, one for each comment provided by the media. Using BERT's pretrained language model, we get a $(n + 1 \times \texttt{tokens} \times 768)$ 3D tensor, where each token is embedded with the 768 dimensions provided by BERT.

Let BERT$[x_0, \ldots, x_n]$ be the tensor constructed using BERT. BERT$[x_0, \ldots, x_n]$ is processed using the Transformer encoder. We use twelve transformer blocks to encode this input, obtaining a new vector encoding Enc$[x_0, \ldots, x_n]$. We combine Enc$[x_0, \ldots, x_n]$ with a social attention encoding before ingesting it into the decoder.

We define our social attention encoding using users' likes. The social attention encoding has $n + 1$ components, one for each piece of text encoded in Enc$[x_0, \ldots, x_n]$. Each attention weight takes values in $[0, 1]$. For the news encoding Enc$[x_0]$, we define a maximum attention weight Att$[x_0] = 1$. For each comment $x_1, \ldots, x_n$, the attention is defined by:

$$\text{Att}[x_i] = \sqrt{\frac{\text{likes}(x_i)}{\text{Max}_{likes}}}, \quad i \in \{1, \ldots, n\},$$

where Max_{likes} is the number of likes recorded by the most salient comment related to x_0. We compose Enc[...] and Att[...] using element-wise dot products. We use the Hadamard product for this purpose, getting a new encoding vector:

$$\text{Enc}_{\text{Att}}[x_i] = \text{Att}[x_i] \odot \text{Enc}[x_i], \quad i \in \{1, \ldots, n\}.$$

The element-wise encoding strategy gives relevance to salient posts, diminishing the importance of comments with few likes. This encoding is then fed into the Transformer decoder to generate a new text. We use six decoder blocks for this task. The transformer uses a language model in its output to map the sequence generated by the decoder to the space of terms considered in the vocabulary. The decoder outcome is obtained by decoding the input provided by our social attention text encoding mechanism:

$$y = \text{Dec}(\text{Enc}_{\text{Att}}[x_0, \ldots, x_n]).$$

Figure 3.1 shows the architecture proposed to solve the summarization tasks. To train this model, we will define a task that simultaneously generates the description of the news and the most salient comments.

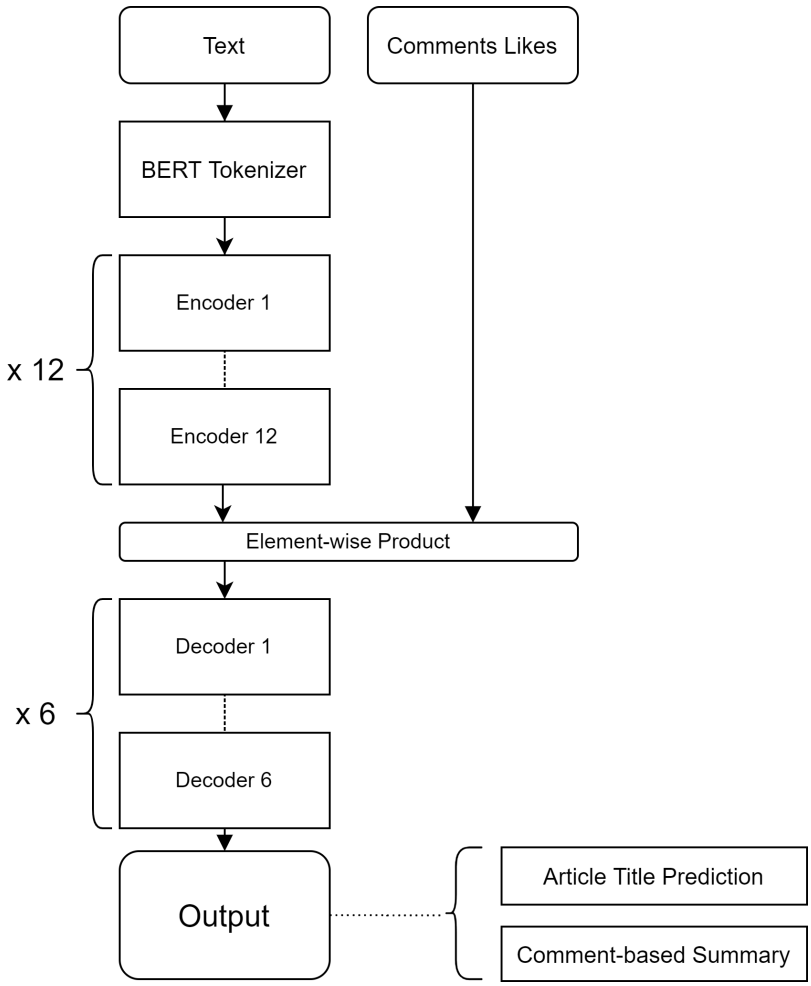


Figure 3.1: High-level model architecture. We use BERT to encode the text. Then, we process these data using the Transformer encoder. Attention weights are combined with text encodings, feeding them into the Transformer decoder.

3.3. Model training

To train the model, we define two tasks that are simultaneously addressed during the training stage. The first task is to generate comments randomly picked from the entry (x_1 to x_n)

according to their attention weights. In this way, the model will learn to generate the most salient comments on each news. The second task considers the generation of the news title. In this way, the model will learn to predict the description of the news. We claim that by solving both tasks simultaneously, the decoder will relate comments and news descriptions.

We encode the training tasks as follows:

$$y = \texttt{<News title> [UNUSED-10] <Chosen Comment (s)>}$$

where y is the target text, and it denotes the text decoded using our proposal. The number of comments included in the target is a parameter of the training stage. We evaluate different settings for this task, considering just one comment or multiple comments.

Note that as the training examples are generated using a random process, we reduce overfitting chances. Also, we increase the number of available examples as we may run many iterations over the training partition producing different examples. More training iterations will produce a greater variety of training examples, improving the generalization capabilities of the model.

For training, we use the standard categorical cross-entropy loss function. We also include label smoothing minimizing the Kullback-Leibler divergence. This way of training the model is similar to the one used in the Open Neural Machine Translation project (Open NMT) [10]. To handle the code generation at the model output, we use Beam Search [8] with five tokens. Each text sequence was decoded until the model outputs an `[EOS]` token. Repeated tri-grams were blocked to prevent the generation of informal expressions [15]. Our full model can be seen on appendix section B, the code is available on a repository on GitHub[4].

3.3.1. Experimental environment

Every model tested was trained ComputeCanada's[5] Béluga Server using 16 cores of a Intel Xeon 6148 Skylake Processor, 32 GB of RAM and a GPU NVidia Tesla V100 SXM2 (16GB).

[4] https://github.com/nachotp/BertCommentSum
[5] `https://www.computecanada.ca/`

We used five hours of computational time for each model checkpoint. Model checkpoints were saved and evaluated every 2000 steps, picking the top-scoring model for each of the training task variants shown in Table 3.1.

ID	Attention encoding	News title	Chosen comment(s)
1	Disabled	Enabled	One comment
2	Disabled	Disabled	One comment
3	Enabled	Enabled	One comment
4	Enabled	Disabled	One comment
5	Disabled	Enabled	Three comments
6	Disabled	Disabled	Three comments
7	Enabled	Enabled	Three comments
8	Enabled	Disabled	Three comments

Table 3.1: Variants of our model evaluated in this study.

As Table 3.1 shows, we trained different variants of our model, enabling and disabling the attention encoding mechanism. We also study different task variants, considering tasks with and without title prediction. For the chosen comments, we study the prediction of one or three comments, all of them randomly picked from the input according to their attention weights.

Tasks addressed in the testing instances. To validate the models in the testing instances, we provide to each model the news title and the reader's comments. Users' likes were not provided to the models. In this way, the task that the model must address consists of, in addition to generating the summary from the text provided, identifying salient comments from the input. Variants with and without news titles were also considered, as indicated in Table 1.

3.4. Validation metrics

To verify the relationship between model's outcomes and testing instances, we evaluate the model's performance computing the ROUGE score between the comments' summary and each comment on the news' thread. Accordingly, we obtain a ROUGE score distribution. We compare the obtained distribution with the distribution of likes received by news' comments. A well-performing model is expected to show similar distributions between likes and ROUGE scores, as it is shown in Figure 3.2.

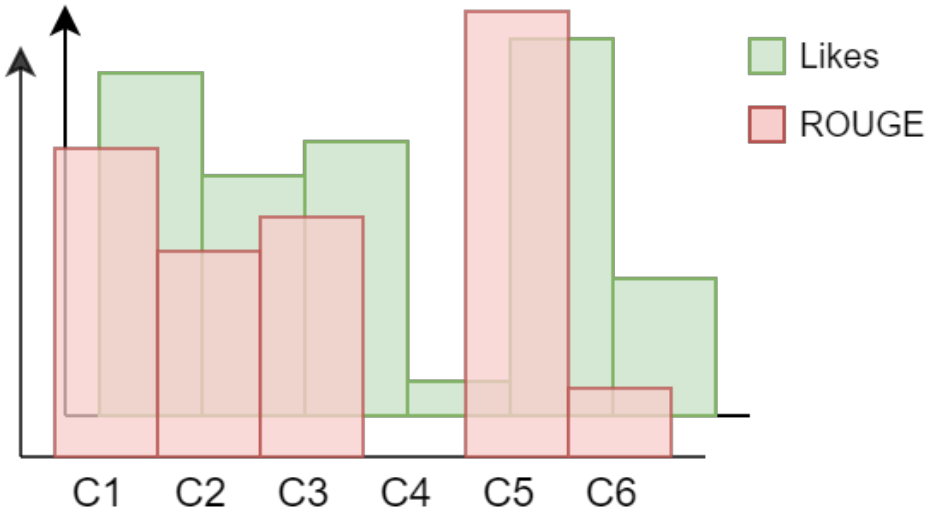


Figure 3.2: Distribution of ROUGE scores and likes obtained from a news item's comments and the text generated by our model. We expect that these distributions correlate, showing that the model can generate texts related to the comments that receive the most likes.

We measure the match between both distributions computing the cross-entropy, which corresponds to a distributional measure named XENT(Rouge). Since XENT(Rouge) is calculated using cross-entropy, better performance corresponds to values close to zero.

In addition to using XENT(Rouge) to evaluate our models, we further validate our results by computing a weighted average recall score of the comments, weighting them according to

the amount of likes they received:

$$Recall_w = \frac{\sum_{i=0}^{n} ROUGE_{recall}(x_i) \cdot likes(x_i)}{\sum likes(x_i)}$$

Furthermore, we calculate the Recall ROUGE score between the news title and the generated summary. By doing so, we obtain performance measures for how relevant is the summary when compared to the discussion topic of the thread.

As we test variants of our model disabling the attention encoding or the title prediction task, we compare the performance achieved with and without the indicated variant using a paired T-test. With these tests, we check if the difference between both variants is statistically significant.

Chapter 4

Results

In this chapter, we evaluate the models' performance for their ability to generate summaries; here, we aggregate results of our validation set according to our XENT(Rouge) Score, Weighted Recall, and Title Rouge Recall. Afterward, we discuss our results by reviewing the best and worst-performing quartiles of our results to find possible reasons that could impact the quality of a generated summary. Following this, we take good and bad examples to show the ability of our model to sample popular comments

4.1. ROUGE Score results

We show in Tables 4.1 and 4.2 the results obtained for predicting one comment or three comments, respectively. We use two state-of-the-art methods to compare the results obtained by our model with its closest competitors. We selected two methods, one extractive and one abstractive. The extractive method corresponds to the centroid-based model proposed by [17]. This method was selected as it allows us to evaluate our model's capacity against one based on pretrained language models with an extractive approach. In this way, we can evaluate our model's contribution to an extractive method built on the same language model. We used BERT to implement the centroid-based model. The other state-of-the-art method selected for the validation section was the abstractive technique proposed by [18]. In this

way, we can evaluate our model's contribution against another considered state of the art but based on the seq2seq architecture.

One comment			
Model	XENT(Rouge)	$Recall_w$	ROUGE
Extractive [17]	0.462	0.515	0.438
Abstractive [18]	0.386	0.594	0.518
Model-(1)	**0.358**	**0.613**	**0.555**
Model-(2)	0.494	0.510	0.458
Model-(3)	0.372	0.582	0.548
Model-(4)	0.395	0.596	0.530

Table 4.1: Average scores for XENT(Rouge), Weighted Recall and mean ROUGE scores over the news article's title prediction, for the single comment prediction task variants.

Three comments			
Model	XENT(Rouge)	$Recall_w$	ROUGE
Extractive [17]	0.178	0.781	0.692
Abstractive [18]	0.187	0.770	0.711
Model-(5)	0.176	0.798	0.734
Model-(6)	0.243	0.727	0.668
Model-(7)	**0.153**	**0.838**	**0.783**
Model-(8)	0.175	0.827	0.779

Table 4.2: Average scores for XENT(Rouge), Weighted Recall and mean ROUGE scores over the news article's title, for the prediction of three comments.

Table 4.1 shows that when generating only one comment and the news title, the best model variant is Model-(1). Model-(1) disables the attention encoding of likes and enables the title prediction task together with the prediction of a salient comment. This variant of the model achieves the best XENT(Rouge) (lowest) and the highest ROUGE score and weighted recall. The other three variants of the model are not effective in this task. Table 4.1 also shows

that our model outperforms both state-of-the-art models. The differences between Model-(1) and the rest of the methods are statistically significant ($p < 0.05$, paired t-test). Table 4.1 also shows that when activating attention encoding, the performance of Model-(3) declines concerning Model-(1) in both metrics. This result means that the attention encoding variant does not contribute relevant information in the news title + single comment generation task. This may be because the model prioritizes the alignment between the title text and the most salient comment, which could be enough to solve this task.

Table 4.2 shows that when generating three comments and the news title, the best model variant is Model-(7). Model-(7) uses attention encoding, making the model obtain the best XENT(Rouge) and the highest ROUGE score. The other three variants of the model are less effective in this task. Table 4.2 also shows that our model outperforms both state-of-the-art models. The differences between Model-(7) and the rest of the methods are statistically significant ($p < 0.05$, paired t-test). The results also show that by activating the attention encoding, it is possible to improve both tasks, significantly reducing the XENT(Rouge) and the ROUGE score concerning Model-(5), trained on the same task but with attention encoding disabled. When disabling the news title prediction task and the attention encoding, the worst result is obtained in XENT(Rouge), which shows that generating many comments requires the attention encoding.

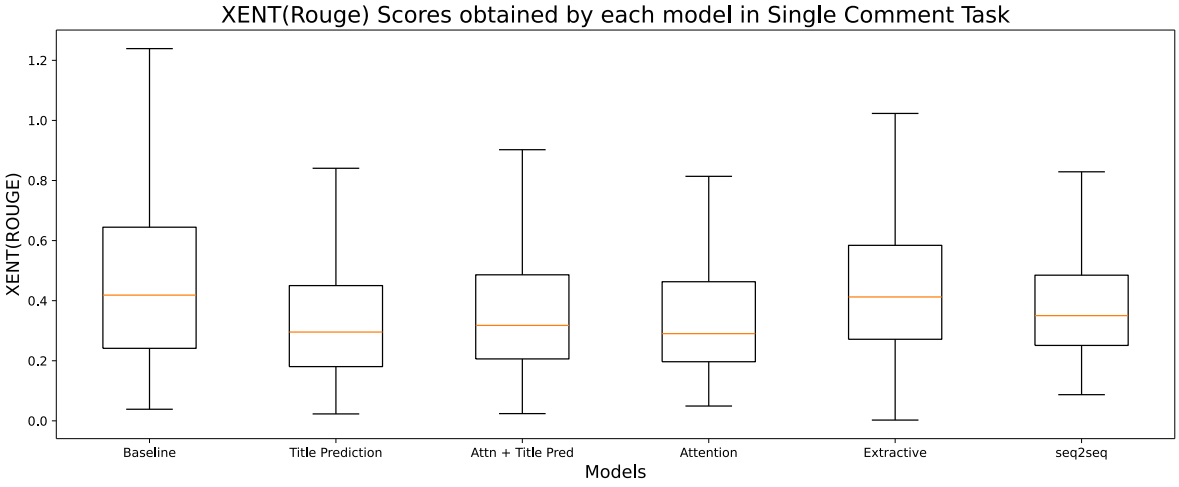


Figure 4.1: Boxplots showing the distribution of XENT(ROUGE) scores obtained by every model and baselines for the Single Comment Prediction task

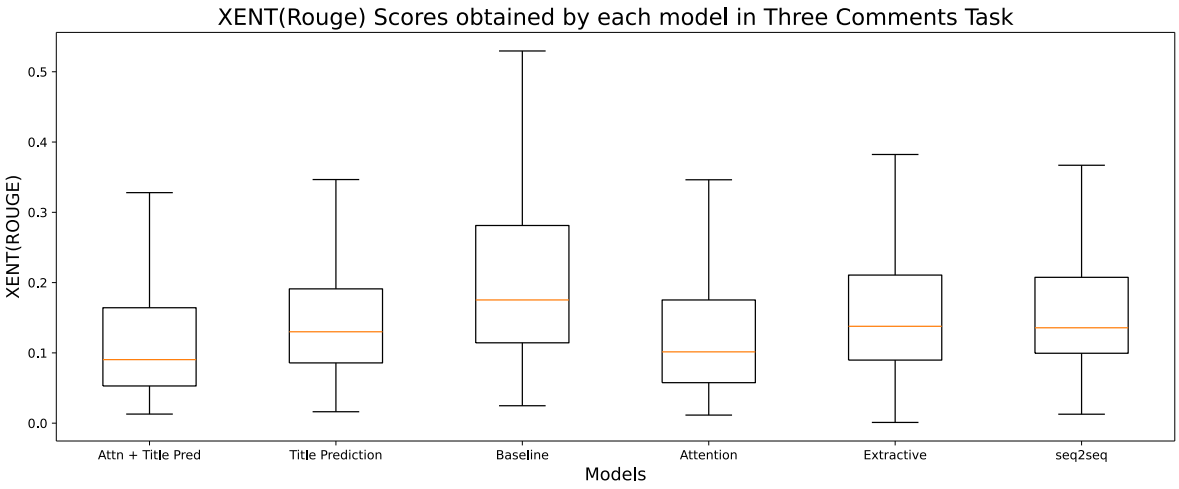


Figure 4.2: Boxplots showing the distribution of XENT(ROUGE) scores obtained by every model and baselines for the Triple Comment Prediction task

Graphs 4.1 and 4.2 show the distribution of scores, here we can see how the baselines have wider ranges for scores while best performing models are more centred towards lower values, this supports our claims of our best performing models. Even though in single Comment Prediction, the best model was the one without attention encoding, that model was close second. This situation changes when we perform triple comment prediction.

4.2. Discussion of results

We characterize the best and worst results of our models. To do this, we partitioned the testing instances into quartiles, with Q1 and Q4 being the quartiles that have the instances with the best and worst XENT(Rouge) scores achieved by our models. To characterize these instances, we measured different characteristics, such as the length of the news thread, the number of comments, and the length of the comment(s) to be generated. We also measure the lexical diversity of the news thread and the salient comment(s). For each quartile, we computed averaged features across testing instances. Finally, we also measure the standard deviation of the distribution of likes. These results are shown for our best models, Model-(1) and Model-(7), in Tables 4.3 and 4.4, respectively.

One comment		
Feature	Q1	Q4
Length of the thread	386	322
Length of the salient comment	84	21
# of comments	21	17
LD of the thread	53%	58%
LD of the salient comment	72%	91%
STD of likes	0.217	0.187

Table 4.3: Characterization of the best and worst results achieved by Model-(1). Testing instances were sorted according to their XENT(Rouge) scores. LD indicates lexical diversity and STD standard deviation.

Table 4.3 shows that shorter threads are more difficult for Model-(1). This characteristic is also expressed in the number of comments. On average, the most difficult instances have 15 comments. The more difficult examples also have a distribution of likes with a lower standard deviation. These examples also show a low lexical use (higher LD), showing that lexical use in these types of examples is also poorer. These trends are also observed when evaluating Model-(7), as it is shown in Table 4.4. For this model, the most difficult instances (Q4) also correspond to shorter news threads, with fewer comments and poorer lexical use. This evaluation also shows that the most difficult instances have a lower standard deviation than those of Q1.

We disaggregate each quartile results according to the number of comments into three groups of testing instances. These results are shown in Tables 4.5 and 4.6 for Model-(1) and Model-(7), respectively. Groups 1, 2, and 3 are folds inside each quartile of the same size in terms of the number of testing instances. Groups indicate instances with fewer or more comments, where G1 is the group with the fewest comments, and G3 is the group with the most comments. Table 4.5 shows that in Q1, the salient comment of G1 is longer than that of groups G2 and G3. This finding indicates that the salient comment is very informative in the case of shorter threads. On the other hand, in longer threads, the salient comment is shorter, which

Three comments		
Feature	Q1	Q4
Length of the thread	404	267
Length of the salient comments	165	68
# of comments	21	14
LD of the thread	52%	60%
LD of the salient comments	63%	76%
STD of likes	0.218	0.179

Table 4.4: Characterization of the best and worst results achieved by Model-(7). Testing instances were sorted according to their XENT(Rouge) scores. LD indicates lexical diversity and STD standard deviation.

indicates that the information tends to be distributed in more comments instead of concentrating on just one. This trend is also observed in Q4, where the lexical diversity of the salient comments is much poorer than in Q1. This finding indicates that in scenarios where lexical diversity is greater, Model-(1) performs better.

When we evaluate the performance of Model-(7) in Table 4.6, we note that the trends observed in Table 4.5 are maintained. Shorter news threads have longer salient comments, and long threads have shorter salient comments. In Q4, we found that Model-(7) has more difficulties generating multi-comments in longer threads. However, in Q1, Model-(7) obtains good performance by generating multi-comments in both long and short threads. This finding reinforces the idea that the main difference between Q1 and Q4 for Model-(7) is the lexical diversity, being that of Q4 much poorer than that found in Q1.

Quartile	Group	Length of the thread	Length of the salient comment	# of comments	LD of the thread	LD of summary	STD of likes	**XENT(Rouge)**
	G1	304	100	12	0.55	0.70	0.21	0.119
Q1	G2	398	89	20	0.53	0.78	0.22	0.113
	G3	400	63	26	0.53	0.88	0.21	0.127
	G1	103	23	6	0.73	0.89	0.14	0.814
Q4	G2	395	22	18	0.52	0.90	0.20	0.693
	G3	398	18	24	0.53	0.93	0.20	0.680

Table 4.5: Disaggregation of results in testing folds according to the number of comments that compound each news thread. Q1 and Q4 indicate the testing instances where Model-(1) performs better and worse, respectively.

Quartile	Group	Length of the thread	Length of the salient comment	# of comments	LD of the thread	LD of the salient comment	STD of likes	**XENT(Rouge)**
	G1	389	219	13	0.52	0.58	0.21	0.035
Q1	G2	408	162	21	0.52	0.63	0.22	0.035
	G3	399	129	28	0.51	0.67	0.22	0.038
	G1	121	66	6	0.69	0.74	0.16	0.371
Q4	G2	367	75	17	0.54	0.76	0.19	0.372
	G3	394	59	24	0.54	0.80	0.20	0.425

Table 4.6: Disaggregation of results in testing folds according to the number of comments that compound each news thread. Q1 and Q4 indicate the testing instances where Model-(7) performs better and worse, respectively.

4.2.1. Illustrative examples

In the following pages, we show some illustrative examples that help in understanding how our model works. We picked four examples from the testing partition, at random, showing the news title, a brief explanation of the context, some illustrative comments, and the social context summarization provided by our model. We show two examples for the single comment task and two examples for the multi comment task. For each of these configurations, we show an example in which our model performs well and another in which it performs poorly. We highlight in magenta the comments to which our model paid more attention to generate the summary. Some less relevant comments are shown in yellow. A portion of the non-relevant comments were left out of the examples. Examples in their original language are available in the Appendix section A.

Examples a) and c) are used in single comment tasks and b) and d) in multi comment tasks. Our model achieves good results in a) and b) and poor results in c) and d). Example a) shows controversial news that produces opposed views among users. The community of this media gives more likes to the comments that support the government and less to those against the government. The most relevant comment corresponds to user 1, who receives 74 likes. The comments of users 3, 4, and 5 also receive likes. The comments of users 2 and 6 are much less relevant to this media community and receive very few likes. When generating the social context summarization, the model extracts text from the most salient comments, generating a long text that summarizes a good part of the users' opinions who support the government. The model discarded comments in yellow. This example shows that attention encoding allows the model to focus on users' most relevant comments, discarding those that record few likes. It can be seen in this example that the comments have lexical richness. The model pays attention to this factor and generates a very informative summary of the discussion. Another successful example of our model is the one shown in b). The generated summary corresponds to a multi comment task. We can also observe that the discussion's lexical richness is important, which is used by the model to generate an informative summary of the discussion. However, unlike the previous example, the model does not pay attention to

News title: *The government supports his Minister and emphasizes that his comments to the Museum do not represent him nowadays*

Context: *A government minister makes statements against the museum of human rights*

Comments:

- User 1 : *And what is the problem that he thinks like that, just like millions of people?* {likes: 74}
- User 2 : *Hey Lone ranger, of what millions are you talking about, when all of our country abhorred HH.RR. violations* {likes: 1}
- User 3 : *It would be valuable for visitors to comprehend, for example, that between many of the factors that contributed to the destruction of democracy, prevailing violence had a significant role.* {likes: 10}
- User 4 : *In June of 2012, the Director of libraries, archives and museums of this government manifested her discomfort with the museum because on her opinion it offers "an incomplete look of the facts" and that it should "contribute to the harmony of society"* {likes: 6}
- User 5 : *⟨Anonymized⟩ is not the first a collaborator of the government that thinks critically of the museum.* {likes: 10}
- User 6 : *It is the museum that is wrong and only shows one of the sides of the coin.* {likes: 2}

Social context summarization: *And what is the problem that he thinks like that, just like millions of people? It would be valuable for the contingency that we live today, it would be a great contribution that the museum explained the situation before the coup, as it would put a note of attention on the limits, the need to take care of democratic forms and procedures. In June of 2012, ⟨anonymized⟩, manifested his discomfort with the museum because on his opinion it offers "an incomplete look of the facts" and that it should "contribute to the harmony of society"*

(a)

News title: *⟨Politician⟩ comments about not presenting in the constitutional Court: "It does not offer any guarantee of legitimacy and I do not go to places where there is no seriousness"*

Context: *A politician comments on his decision not to attend court*

Comments:

- User 1 : *I believed him more when he was an "objective" journalist.* {likes: 10}
- User 2 : *The ⟨anonymized⟩ constitutional Court, inspiration for ours, is completely politicized, it has been one of the reasons for the ⟨anonymized⟩ government to want independence from ⟨anonymized⟩.* {likes: 10}
- User 3 : *In our country is no different* {likes: 18}
- User 4 : *It's a political body not legal.* {likes: 40}
- User 5 : *What is this man talking about, the constitutional reform of 2005 says that the constitutional court will be composed by 3 members chosen by the supreme court in secret ...* {likes: 29}
- User 6 : *Beyond the respect that the members of the constitutional court deserve, I think it would be positive if the process for naming its members is modified by establishing a mechanism that does not involve so much interference from political power...* {likes: 1}
- User 7 : *He disqualifies one of the members of the constitutional Court because offer any guarantee of legitimacy or seriousness, ...* {likes: 1}

Social context summarization: *What is this man talking about, the constitutional reform of 2005 says that the constitutional court will be composed by 3 members chosen by the supreme court in secret... Beyond the respect that the members of the constitutional court deserve, I believe to change the process for naming its members ... He disqualifies one of the members of the constitutional Court because offer any guarantee of legitimacy or seriousness... It's a political body it has no program.*

(b)

News title: *The president assists to a "class" about cybersecurity in Singapore and visits a water treatment plant*

Context: *The president's agenda raises suspicions about the purpose of his visit.*

Comments:

- User 1 : *Did he already show the piece of paper of the ⟨anonymized⟩?* {likes: 3}
- User 2 : *Be careful president, do not go to drink water with poop* {likes: 1}
- User 3 : *Is he doing personal business as he did in ⟨country⟩?* {likes: 1}
- User 5 : *He should attend to ethics and morality classes* {likes: 5}

Social context summarization: *He should attend to ethics and morality classes. To classes, classes of ethical and morality he should attend he should.*

(c)

News title: *Number of missing people from floods in ⟨place⟩ decreased from 18 to 8*

Context: *This news reduces the initial number of missing persons in a flood.*

Comments:

- User 1 : *There is too much mixed river to focus on details of that level.* {likes: 2}
- User 2 : *(Not "⟨foreseee⟩" like others have said ⟨may⟩ times)* {likes: 2}
- User 3 : *Collaborate and support as long as it means helping.* {likes: 2}
- User 4 : *At what time will the water arrive!* {likes: 2}
- User 5 : *⟨location⟩, at the foot of the ⟨location⟩ Volcano* {likes: 2}
- User 6 : *I hope the competent authorities take advantage of this new experience.* {likes: 2}

Social context summarization: *⟨location⟩, at the foot of the ⟨location⟩ Volcano. You should learn more about your province. At what time will the water arrive! Refugees too much mixed river to focus on details of that level. Not "⟨Foreseee⟩", like others have said ⟨may⟩ times).*

(d)

some comments with likes, such as comments in yellow. This is because user 1's comment is about the politician, not about the subject of the news. Something similar happens with the comments of users 2 and 3, who compare the context of the news with the situation observed in another country. These three comments have in common that they do not refer directly to the news subject but to tangential issues. The comments of users 4, 5, 6, and 7 refer to the news subject and therefore are more relevant when generating the summary. In this example, the attention encoding was not so relevant for the model, but rather the comments' alignment with the news title. This finding relieves the fact that our model obtains its best result when solving both tasks simultaneously: news title generation and multi comment generation. Solving both tasks simultaneously helps the model to find comments that are aligned with the subject of the news. Example c) shows an unfavourable case for our model. This example has a poor lexical diversity and, at the same time, is very uninformative in terms of likes. The model, in this example, only takes one comment to produce an uninformative summary. It is also observed that in this example, the model uses a convoluted grammar, giving two alternative versions of the same sentence. The discussion is poor since it does not refer directly to the news subject but rather to the President. In this scenario, the model does not find how to align the comments with the news subject and generates an uninformative summary. Something similar happens with example d). This example is uninformative in terms of likes. In turn, the news subject is not very descriptive, and therefore it isn't easy to align it with the comments. The model pays attention to four comments, all of which produce inconsistent text.

4.2.2. Main findings and limitations of our model

Experimental results show that our model behaves well in cases where the news discussion has lexical richness. We have also observed that, in general, shorter threads have longer salient comments and longer threads have shorter salient comments. This finding suggests that there is an invariant in terms of information distribution in social discussions. Our data show that there are discussions with very little lexical richness and others with high lexical richness. The proposed model takes advantage of the lexical richness to generate an informative summary. If the discussion is poor, so is the summary.

Likes encoding allows the model to prioritize the comments with the most likes. This mechanism is successful if the discussion is lexically rich. If the discussion is lexically poor, the model is unable to generate an adequate summary. This is because the model has limitations in relating one discussion to other discussions. The model operates by constructing a summary by paying close attention to the news's comments but little attention to the comments associated with similar news, while comparing Single versus Triple comment generation, we can see that the attention encoding becomes much more reliable. The model also shows some risks. In cases where the news produces controversial points of view, the attention encoding mechanism prioritizes a view, the one with the most likes, discarding comments from minorities. This bias towards the most voted comments can introduce an over-representation of the dominant points of view, minimizing the influence of minority groups that contribute with a greater diversity of opinions to the discussion. The generation of summaries of social contexts that do not amplify these biases is a relevant challenge for this model type.

Chapter 5

Conclusions

This research aimed to generate abstractive summaries of online conversations by including the social context where this conversations unfold. This task presented many challenges to tackle, from data processing, design of the model's architecture and validation of results.

Unsupervised text summarization allows us to extend the range of settings where standard text summarization cannot work due to the lack of human-made golden summaries, while it usually cannot compete in readability and correctness, saliency of information can be achieved using these models. The use of user comments to generate these standards brought it's own challenges such as slang abuse and lexical typos to the summarization task. We have developed a transformer-based model that allows us to summarize news articles' comments while considering the social context that these interactions live in, and is able to extend the pretrained language model with the slang used by people by fine-tunning the sub-word tokenization embedding.

We compare our model with state-of-the-art methods abstractive and extractive methods. Our experimental results show that our model performs better than its competitors as it can take advantage of the inferred popularity of comments when trained with attention encoding. In particular, our model can take advantage of discussions when they are lexically rich, sampling several comments to maximize recall of infornmation.

Chapter 6

Future Work

Future work aims to handle some of the limitations discussed. particularly the coverage of different opinions that do not necessarily represent the majority.This can be achieved by processing the data to detect diffrent opinion clusters, these could extend our training task to promote diversity in comments by chosing popular comments of each group instead of the whole thread, leading to a more balanced view of the users' opinions on social platforms even if these are not as popular.

Additionally, this work reveals the need to develop validation metrics to address ROUGE's limitations on Unsupervised summarization, our approach focused on measuring the summary's ability to retrieve information proportional to inferred popularity but there is no penalization for repetition or poorly written text.

Bibliography

[1] Almahy, Ibrahim y Naomie Salim: *Web discussion summarization: study review*. En *Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013), December 16-18, 2013, Kuala Lumpur, Malaysia*, páginas 649–656. Springer, 2013.

[2] Cañete, José, Gabriel Chaperon, Rodrigo Fuentes y Jorge Pérez: *Spanish Pre-Trained BERT Model and Evaluation Data*. En *PML4DC co-located at the 8th International Conference on Learning Representations, ICLR 2020, April 26-30, 2020, Addis-Ababa, Ethiopia*, 2020.

[3] Chen, Yen-Chun y Mohit Bansal: *Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting*. En *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, páginas 675–686, 2018.

[4] Devlin, Jacob, Ming-Wei Chang, Kenton Lee y Kristina Toutanova: *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. En *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, páginas 4171–4186, 2019.

[5] Ding, Ying y Jing Jiang: *Towards Opinion Summarization from Online Forums*. En *Recent Advances in Natural Language Processing, RANLP 2015, 7-9 September, 2015, Hissar, Bulgaria*, páginas 138–146, 2015.

[6] Gao, Shen, Xiuying Chen, Piji Li, Zhaochun Ren, Lidong Bing, Dongyan Zhao y Rui Yan: *Abstractive text summarization by incorporating reader comments*. En *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, January 27 - February 1, 2019, Honolulu, Hawaii, USA*, volumen 33, páginas 6399–6406, 2019.

[7] Jadhav, Aishwarya y Vaibhav Rajan: *Extractive Summarization with SWAP-NET: Sentences and Words from Alternating Pointer Networks*. En *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, páginas 142–151, 2018.

[8] Johnson, Melvin, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes y Jeffrey Dean: *Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation*. Transactions of the Association of Computer Linguistics, TACL, 5:339–351, 2017.

[9] Kim, Byeongchang, Hyunwoo Kim y Gunhee Kim: *Abstractive Summarization of Reddit Posts with Multi-level Memory Networks*. En *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, páginas 2519–2531, 2019.

[10] Klein, Guillaume, Yoon Kim, Yuntian Deng, Jean Senellart y Alexander Rush: *OpenNMT: Open-Source Toolkit for Neural Machine Translation*. En *Proceedings of ACL 2017, System Demonstrations*, páginas 67–72, Vancouver, Canada, Julio 2017.

[11] Kryscinski, Wojciech, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong y Richard Socher: *Neural text summarization: A critical evaluation*. En *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, November 3, 2019, Hong Kong, China*, páginas 540–551, 2019.

[12] Li, Quanzhi y Qiong Zhang: *Abstractive Event Summarization on Twitter*. En *Proceedings of the Web Conference, WWW 2020, April 20-24, 2020, Taipei, Taiwan*, página 22–23, 2020.

[13] Lin, Chin Yew: *ROUGE: A Package for Automatic Evaluation of Summaries*. En *Text Summarization Branches Out*, páginas 74–81, Barcelona, Spain, Julio 2004. Association for Computational Linguistics.

[14] Lin, Junyang, Xu Sun, Shuming Ma y Qi Su: *Global Encoding for Abstractive Summarization*. En *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, páginas 163–169, 2018.

[15] Liu, Yang y Mirella Lapata: *Text Summarization with Pretrained Encoders*. En *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, páginas 3728–3738, 2019.

[16] Ng, Jun Ping y Viktoria Abrecht: *Better summarization evaluation with word embeddings for rouge*. arXiv preprint arXiv:1508.06034, 2015.

[17] Rossiello, Gaetano, Pierpaolo Basile y Giovanni Semeraro: *Centroid-based Text Summarization through Compositionality of Word Embeddings*. En *Proceedings of the Workshop on Summarization and Summary Evaluation Across Source Types and Genres, MultiLing@EACL 2017, Valencia, Spain, April 3, 2017*, páginas 12–21, 2017.

[18] Shi, Tian, Yaser Keneshloo, Naren Ramakrishnan y Chandan K Reddy: *Neural abstractive text summarization with sequence-to-sequence models*. arXiv preprint arXiv:1812.02303, 2018.

[19] Subramanian, Sandeep, Raymond Li, Jonathan Pilault y Christopher Pal: *On extractive and abstractive neural document summarization with transformer language models*. arXiv preprint arXiv:1909.03186, 2019.

[20] Sutskever, Ilya, Oriol Vinyals y Quoc V. Le: *Sequence to Sequence Learning with Neural Networks*. En *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, páginas 3104–3112, 2014.

[21] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser y Illia Polosukhin: *Attention is All you Need*. En *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, páginas 5998–6008, 2017.

[22] Wang, Li, Junlin Yao, Yunzhe Tao, Li Zhong, Wei Liu y Qiang Du: *A Reinforced Topic-Aware Convolutional Sequence-to-Sequence Model for Abstractive Text Summarization*. En *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, páginas 4453–4460, 2018.

[23] Zhou, Qingyu, Nan Yang, Furu Wei y Ming Zhou: *Sequential Copying Networks*. En *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, AAAI-18, February 2-7, 2018, New Orleans, Louisiana, USA*, páginas 4987–4995, 2018.

Appendices

A. Original Illustrative Examples

Original examples in Spanish from the results, each corresponds with the letter of it's translated version.

B. Full Model

```
AbsSummarizer(
  (bert): Bert(
    (model): BertModel(
      (embeddings): BertEmbeddings(
        (word_embeddings): Embedding(31002, 768, padding_idx=0)
        (position_embeddings): Embedding(512, 768)
        (token_type_embeddings): Embedding(2, 768)
        (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
        (dropout): Dropout(p=0.1, inplace=False)
      )
      (encoder): BertEncoder(
        (layer): ModuleList(
          (0): BertLayer(
            (attention): BertAttention(
              (self): BertSelfAttention(
                (query): Linear(in_features=768, out_features=768, bias=True)
```

News title: *Gobierno respalda a rojas y enfatiza que ese planteamiento sobre el museo de la memoria hoy día no lo representa*

Comments:

- User 1 : *y cuál es el problema de que piense así, igual que millones de chilenos?* {likes: 74}
- User 2 : *Oye llanero solitario de cuales millones hablas, si todo chile aborrecio las violaciones de los ddhh.* {likes: 1}
- User 3 : *sería valioso que sus visitantes comprendiesen, por ejemplo, que entre muchos de los factores que contribuyeron a la destrucción de la democracia, tuvo un rol significativo la violencia imperante.* {likes: 10}
- User 4 : *en junio de 2012, magdalena krebs, archivosymuseos "directora de bibliotecas, archivos y museosdurante el gobierno de sebastián piñera, manifestó su disconformidad con el museo que a su parecer ofrece "una visión incompleta de los hechos" y que éste debiera "contribuir a la armonía de la sociedad ".* {likes: 6}
- User 5 : *rojas no es elprimer colaborador de piñera que opina críticamente del museo.* {likes: 10}
- User 6 : *es el museo el que está mal planteado y sólo muestra una de las caras de la moneda.* {likes: 2}

Social context summarization: *y cuál es el problema de que piense así, igual que millones de chilenos ? sería valioso la contingencia que vivimos hoy, sería una gran contribución que el museo explicara los hechos anteriores al golpe, pues pondría una nota de atención sobre los límites, la necesidad de cuidar las formas y los procedimientos democráticos. en junio de 2012, gonzalo blumel, manifestó su disconformidad con el museo que a su parecer ofrece "una visión incompleta de los hechos" y que éste debiera "contribuir a la armonía de la sociedad"*

(a)

News title: *Guillier por no exponer en el TC: "no ofrece ninguna garantía de legitimidad y yo no voy a lugares donde no hay seriedad"*

Comments:

- User 1 : *Le creia mas cuando era un periodista "objetivo".* {likes: 10}
- User 2 : *El tribunal constitucional español, inspiración del nuestro, está totalmente politizado, ha sido una de las causales para que el gobierno catalán quiera la independencia de españa.* {likes: 10}
- User 3 : *en chile no es diferente.* {likes: 18}
- User 4 : *es un órgano político no jurídico.* {likes: 40}
- User 5 : *pero de que habla este señor, si según la reforma constitucional del 2005 dice que el tc estará compuesto por3 miembros elegidos por la corte suprema en votación secreta3 miembros designados por el presidente de la república 2 miembros elegidos por el senado (por los de los miembros en ejercicio) 2 miembros propuestos por la cámara de diputadosy confirmados por el senado (por los de los miembros en ejercicio, en ambas instancias).* {likes: 29}
- User 6 : *más allá del respeto que me merecen los miembros del tc, creo que sería positivo que se modifique el proceso de nombramiento de sus integrantes estableciendo un mecanismo que no implique tanta injerencia del poder politico, lo mismo para el nombramiento de los ministros de la corte suprema.* {likes: 1}
- User 7 : *primero descalifica a los integrantes del tc por queno ofrecen ninguna garantía de legitimidad ni seriedad, primero porque se cuotean políticamente y no representan lo mejor de nuestra intelectualidad ética, jurídica y académica.* {likes: 1}

Social context summarization: *pero de que habla este señor, si según la reforma constitucional del 2005 dice que el tc estará compuesto por3 miembros elegidos por la corte suprema en votación secreta3 miembros designados por el senado (por los de los miembros en ejercicio) de la cámara de diputadosy confirmados por el presidente de la república 2 miembros en votación secreta 3 miembros nombrados por el congreso (por el gobierno, en ambas instancias). el tc no es un poder del estado. más allá del respeto que me merecen los miembros del tc, creo que se modifique el proceso de nombramiento de sus integrantes estableciendo un mecanismo que no implique tanta injerencia del poder politico, lo mismo para el nombramiento de los ministros de la corte supremo. el tc no se entiende, que quiere o que dice guillier? descalifica a uno de los integrantes del tc por queno ofrecen ninguna garantía de legitimidad ni seriedad, primero porque se cuotean políticamente y no representan lo mejor de nuestra intelectualidad ética, jurídica y académica y académica. es un órgano político político no tiene ningun programa*

(b)

News title: *Piñera asiste a "clase" de ciberseguridad en Singapur y visita planta de tratamiento de agua*

Comments:

- User 1 : *¿ya mostró el papelito de los 33?* {likes: 3}
- User 2 : *cuidado presidente no vaya a tomar agua con caca* {likes: 1}
- User 3 : *anda haciendo negocios personales como lo hizo en argentina?* {likes: 1}
- User 5 : *a clases de etica y moral debiera acudir.* {likes: 5}

Social context summarization: *a clases de etica y moral debiera acudir. a clases, clases de etico y morales debiera asistir debiera*

(c)

News title: *Cifra de desaparecidos por aluviones en Cajón del Maipo bajó de 18 a 8*

Context: *This news reduces the initial number of missing persons in a flood.*

Comments:

- User 1 : *ya hay mucho río revuelto para fijarse en detalles de esa clase* . {likes: 2}
- User 2 : *(no "preveer", como lo han dicho varas veces).* {likes: 2}
- User 3 : *sostener, colaborar y apoyar mientras signifique ayudar.* {likes: 2}
- User 4 : *a que horas va llegar el agua!* {likes: 2}
- User 5 : *refugio plantat, valle de la engorda, a los pies del volcán san josé.* {likes: 2}
- User 6 : *ojala las autoridades competentes saquen provecho de esta nueva experiencia.* {likes: 2}

Social context summarization: *refugio plantat, valle de la engorda, a los pies del volcán san josé. si la intendenta lo nombró así, debería informarse más sobre su provincia. a que horas va llegar el agua! refugios mucho río revuelto para fijarse en detalles de esa clase. no "preveer", como lo han dicho varas veces).*

(d)

```
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(1): BertLayer(
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
```

```
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(2): BertLayer(
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(3): BertLayer(
```

```
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(4): BertLayer(
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
```

```
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(5): BertLayer(
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
```

```
  )
)
(6): BertLayer(
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(7): BertLayer(
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
```

```
      (output): BertSelfOutput(
        (dense): Linear(in_features=768, out_features=768, bias=True)
        (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
        (dropout): Dropout(p=0.1, inplace=False)
      )
    )
    (intermediate): BertIntermediate(
      (dense): Linear(in_features=768, out_features=3072, bias=True)
    )
    (output): BertOutput(
      (dense): Linear(in_features=3072, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (8): BertLayer(
    (attention): BertAttention(
      (self): BertSelfAttention(
        (query): Linear(in_features=768, out_features=768, bias=True)
        (key): Linear(in_features=768, out_features=768, bias=True)
        (value): Linear(in_features=768, out_features=768, bias=True)
        (dropout): Dropout(p=0.1, inplace=False)
      )
      (output): BertSelfOutput(
        (dense): Linear(in_features=768, out_features=768, bias=True)
        (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
        (dropout): Dropout(p=0.1, inplace=False)
      )
    )
    (intermediate): BertIntermediate(
      (dense): Linear(in_features=768, out_features=3072, bias=True)
    )
    (output): BertOutput(
```

```
      (dense): Linear(in_features=3072, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (9): BertLayer(
    (attention): BertAttention(
      (self): BertSelfAttention(
        (query): Linear(in_features=768, out_features=768, bias=True)
        (key): Linear(in_features=768, out_features=768, bias=True)
        (value): Linear(in_features=768, out_features=768, bias=True)
        (dropout): Dropout(p=0.1, inplace=False)
      )
      (output): BertSelfOutput(
        (dense): Linear(in_features=768, out_features=768, bias=True)
        (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
        (dropout): Dropout(p=0.1, inplace=False)
      )
    )
    (intermediate): BertIntermediate(
      (dense): Linear(in_features=768, out_features=3072, bias=True)
    )
    (output): BertOutput(
      (dense): Linear(in_features=3072, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (10): BertLayer(
    (attention): BertAttention(
      (self): BertSelfAttention(
        (query): Linear(in_features=768, out_features=768, bias=True)
        (key): Linear(in_features=768, out_features=768, bias=True)
```

```
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
    (dense): Linear(in_features=768, out_features=3072, bias=True)
  )
  (output): BertOutput(
    (dense): Linear(in_features=3072, out_features=768, bias=True)
    (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
    (dropout): Dropout(p=0.1, inplace=False)
  )
)
(11): BertLayer(
  (attention): BertAttention(
    (self): BertSelfAttention(
      (query): Linear(in_features=768, out_features=768, bias=True)
      (key): Linear(in_features=768, out_features=768, bias=True)
      (value): Linear(in_features=768, out_features=768, bias=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (output): BertSelfOutput(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
  )
  (intermediate): BertIntermediate(
```

```
            (dense): Linear(in_features=768, out_features=3072, bias=True)
          )
          (output): BertOutput(
            (dense): Linear(in_features=3072, out_features=768, bias=True)
            (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
      )
    )
    (pooler): BertPooler(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (activation): Tanh()
    )
  )
)
(decoder): TransformerDecoder(
  (embeddings): Embedding(31002, 768, padding_idx=0)
  (pos_emb): PositionalEncoding(
    (dropout): Dropout(p=0.2, inplace=False)
  )
  (transformer_layers): ModuleList(
    (0): TransformerDecoderLayer(
      (self_attn): MultiHeadedAttention(
        (linear_keys): Linear(in_features=768, out_features=768, bias=True)
        (linear_values): Linear(in_features=768, out_features=768, bias=True)
        (linear_query): Linear(in_features=768, out_features=768, bias=True)
        (softmax): Softmax(dim=-1)
        (dropout): Dropout(p=0.2, inplace=False)
        (final_linear): Linear(in_features=768, out_features=768, bias=True)
      )
      (context_attn): MultiHeadedAttention(
        (linear_keys): Linear(in_features=768, out_features=768, bias=True)
```

```
      (linear_values): Linear(in_features=768, out_features=768, bias=True)
      (linear_query): Linear(in_features=768, out_features=768, bias=True)
      (softmax): Softmax(dim=-1)
      (dropout): Dropout(p=0.2, inplace=False)
      (final_linear): Linear(in_features=768, out_features=768, bias=True)
    )
    (feed_forward): PositionwiseFeedForward(
      (w_1): Linear(in_features=768, out_features=2048, bias=True)
      (w_2): Linear(in_features=2048, out_features=768, bias=True)
      (layer_norm): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
      (dropout_1): Dropout(p=0.2, inplace=False)
      (dropout_2): Dropout(p=0.2, inplace=False)
    )
    (layer_norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
    (layer_norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
    (drop): Dropout(p=0.2, inplace=False)
  )
  (1): TransformerDecoderLayer(
    (self_attn): MultiHeadedAttention(
      (linear_keys): Linear(in_features=768, out_features=768, bias=True)
      (linear_values): Linear(in_features=768, out_features=768, bias=True)
      (linear_query): Linear(in_features=768, out_features=768, bias=True)
      (softmax): Softmax(dim=-1)
      (dropout): Dropout(p=0.2, inplace=False)
      (final_linear): Linear(in_features=768, out_features=768, bias=True)
    )
    (context_attn): MultiHeadedAttention(
      (linear_keys): Linear(in_features=768, out_features=768, bias=True)
      (linear_values): Linear(in_features=768, out_features=768, bias=True)
      (linear_query): Linear(in_features=768, out_features=768, bias=True)
      (softmax): Softmax(dim=-1)
      (dropout): Dropout(p=0.2, inplace=False)
      (final_linear): Linear(in_features=768, out_features=768, bias=True)
```

```
  )
  (feed_forward): PositionwiseFeedForward(
    (w_1): Linear(in_features=768, out_features=2048, bias=True)
    (w_2): Linear(in_features=2048, out_features=768, bias=True)
    (layer_norm): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
    (dropout_1): Dropout(p=0.2, inplace=False)
    (dropout_2): Dropout(p=0.2, inplace=False)
  )
  (layer_norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
  (layer_norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
  (drop): Dropout(p=0.2, inplace=False)
)
(2): TransformerDecoderLayer(
  (self_attn): MultiHeadedAttention(
    (linear_keys): Linear(in_features=768, out_features=768, bias=True)
    (linear_values): Linear(in_features=768, out_features=768, bias=True)
    (linear_query): Linear(in_features=768, out_features=768, bias=True)
    (softmax): Softmax(dim=-1)
    (dropout): Dropout(p=0.2, inplace=False)
    (final_linear): Linear(in_features=768, out_features=768, bias=True)
  )
  (context_attn): MultiHeadedAttention(
    (linear_keys): Linear(in_features=768, out_features=768, bias=True)
    (linear_values): Linear(in_features=768, out_features=768, bias=True)
    (linear_query): Linear(in_features=768, out_features=768, bias=True)
    (softmax): Softmax(dim=-1)
    (dropout): Dropout(p=0.2, inplace=False)
    (final_linear): Linear(in_features=768, out_features=768, bias=True)
  )
  (feed_forward): PositionwiseFeedForward(
    (w_1): Linear(in_features=768, out_features=2048, bias=True)
    (w_2): Linear(in_features=2048, out_features=768, bias=True)
    (layer_norm): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
```

```
    (dropout_1): Dropout(p=0.2, inplace=False)
    (dropout_2): Dropout(p=0.2, inplace=False)
  )
  (layer_norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
  (layer_norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
  (drop): Dropout(p=0.2, inplace=False)
)
(3): TransformerDecoderLayer(
  (self_attn): MultiHeadedAttention(
    (linear_keys): Linear(in_features=768, out_features=768, bias=True)
    (linear_values): Linear(in_features=768, out_features=768, bias=True)
    (linear_query): Linear(in_features=768, out_features=768, bias=True)
    (softmax): Softmax(dim=-1)
    (dropout): Dropout(p=0.2, inplace=False)
    (final_linear): Linear(in_features=768, out_features=768, bias=True)
  )
  (context_attn): MultiHeadedAttention(
    (linear_keys): Linear(in_features=768, out_features=768, bias=True)
    (linear_values): Linear(in_features=768, out_features=768, bias=True)
    (linear_query): Linear(in_features=768, out_features=768, bias=True)
    (softmax): Softmax(dim=-1)
    (dropout): Dropout(p=0.2, inplace=False)
    (final_linear): Linear(in_features=768, out_features=768, bias=True)
  )
  (feed_forward): PositionwiseFeedForward(
    (w_1): Linear(in_features=768, out_features=2048, bias=True)
    (w_2): Linear(in_features=2048, out_features=768, bias=True)
    (layer_norm): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
    (dropout_1): Dropout(p=0.2, inplace=False)
    (dropout_2): Dropout(p=0.2, inplace=False)
  )
  (layer_norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
  (layer_norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
```

```
  (drop): Dropout(p=0.2, inplace=False)
)
(4): TransformerDecoderLayer(
  (self_attn): MultiHeadedAttention(
    (linear_keys): Linear(in_features=768, out_features=768, bias=True)
    (linear_values): Linear(in_features=768, out_features=768, bias=True)
    (linear_query): Linear(in_features=768, out_features=768, bias=True)
    (softmax): Softmax(dim=-1)
    (dropout): Dropout(p=0.2, inplace=False)
    (final_linear): Linear(in_features=768, out_features=768, bias=True)
  )
  (context_attn): MultiHeadedAttention(
    (linear_keys): Linear(in_features=768, out_features=768, bias=True)
    (linear_values): Linear(in_features=768, out_features=768, bias=True)
    (linear_query): Linear(in_features=768, out_features=768, bias=True)
    (softmax): Softmax(dim=-1)
    (dropout): Dropout(p=0.2, inplace=False)
    (final_linear): Linear(in_features=768, out_features=768, bias=True)
  )
  (feed_forward): PositionwiseFeedForward(
    (w_1): Linear(in_features=768, out_features=2048, bias=True)
    (w_2): Linear(in_features=2048, out_features=768, bias=True)
    (layer_norm): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
    (dropout_1): Dropout(p=0.2, inplace=False)
    (dropout_2): Dropout(p=0.2, inplace=False)
  )
  (layer_norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
  (layer_norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
  (drop): Dropout(p=0.2, inplace=False)
)
(5): TransformerDecoderLayer(
  (self_attn): MultiHeadedAttention(
    (linear_keys): Linear(in_features=768, out_features=768, bias=True)
```

```
          (linear_values): Linear(in_features=768, out_features=768, bias=True)
          (linear_query): Linear(in_features=768, out_features=768, bias=True)
          (softmax): Softmax(dim=-1)
          (dropout): Dropout(p=0.2, inplace=False)
          (final_linear): Linear(in_features=768, out_features=768, bias=True)
        )
        (context_attn): MultiHeadedAttention(
          (linear_keys): Linear(in_features=768, out_features=768, bias=True)
          (linear_values): Linear(in_features=768, out_features=768, bias=True)
          (linear_query): Linear(in_features=768, out_features=768, bias=True)
          (softmax): Softmax(dim=-1)
          (dropout): Dropout(p=0.2, inplace=False)
          (final_linear): Linear(in_features=768, out_features=768, bias=True)
        )
        (feed_forward): PositionwiseFeedForward(
          (w_1): Linear(in_features=768, out_features=2048, bias=True)
          (w_2): Linear(in_features=2048, out_features=768, bias=True)
          (layer_norm): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
          (dropout_1): Dropout(p=0.2, inplace=False)
          (dropout_2): Dropout(p=0.2, inplace=False)
        )
        (layer_norm_1): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
        (layer_norm_2): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
        (drop): Dropout(p=0.2, inplace=False)
      )
    )
    (layer_norm): LayerNorm((768,), eps=1e-06, elementwise_affine=True)
  )
  (generator): Sequential(
    (0): Linear(in_features=768, out_features=31002, bias=True)
    (1): LogSoftmax()
  )
)
```