Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Automation and Applied Informatics

Controllable Text Generation in Abstractive Summarization

Bachelor's Thesis

Author
Milán Konor Nyist

Advisor
Judit Ács

December 6, 2023

Contents

HALLGATÓI NYILATKOZAT

Alulírott *Nyist Milán Konor*, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2023. december 6.

Nyist Milán Konor
hallgató

Kivonat

A hatalmas szöveges adatok egyre szélesebb körű hozzáférhetősége szükségessé teszi a hatékony információkinyerés fejlett technikáit. Az absztraktív szövegkivonatolás ebben az összefüggésben döntő szerepet játszik, mivel a hosszú szövegeket tömör, informatív összefoglalókká sűríti. Ez a módszer arra összpontosít, hogy a forrásszövegből csak a lényeges tartalmat emelje ki. Olyan új mondatokat is tartalmazhat, amelyek esetleg nem szerepelnek az eredeti szövegben. Az elfogadható eredmények eléréséhez fontos a kompetens modell, de a megfelelő generálási stratégiák használata is döntő fontosságú az elvárt eredmények eléréséhez. Itt jönnek a képbe a dekódolási stratégiák. A kimeneti tokenek kiválasztásának folyamatát a szöveg generálásához dekódolásnak nevezzük, és a megfelelő stratégia megfelelő körülményekhez való kiválasztása növelheti a modell sikerességi arányát. A modelleket korlátozások által kell irányítani. Ezek bevezetése segíthet elkerülni az olyan tokeneket, amelyek nem feltétlenül megfelelőek kimenetként. Bizonyos esetekben ki akarjuk kényszeríteni, hogy a tokenek jelen legyenek a kimenetünkben. Az irányítható szöveggenerálás témája különböző felhasználási eseteket ölel fel. A HunSum-1[2] kész modell segítségével a dekódolási stratégiák tesztelhetők és értékelhetők emberi értékeléssel és automatizált metrikákkal, például ROUGE[7] és BLEU[12] pontszámokkal.

Abstract

The increasing availability of vast textual data necessitates advanced techniques for efficient information extraction. Abstractive summarization plays a crucial role in this context by condensing lengthy texts into concise, informative summaries. This method focuses on extracting only the essential concepts from the source text. It can include new sentences that may not be present in the original text. It is important to have a competent model in order to reach acceptable results, but using the right generational strategies is crucial in achieving the expected outputs. This is where decoding strategies come into play. The process of selecting output tokens to generate text is known as decoding, and choosing the right strategy for the right circumstance can increase the model's success rate. The models have to be guided by constraints. Introducing these can help us avoid tokens that may not be appropriate as an output. In some cases, we want to force the tokens to be present in our output. The topic of controllable text generation encompasses various use cases from here. Using the ready-made model HunSum-1[2], decoding strategies can be tested and assessed with human evaluation and automated metrics such as ROUGE[7] and BLEU[12] scores.

Chapter 1

Introduction

Natural Language Processing (NLP) has emerged as a groundbreaking field at the intersection of computer science, artificial intelligence, and linguistics, fundamentally changing the way humans interact with machines and information. It is essential to follow the historical trajectory of NLP's development and understand the significant milestones that have brought us to the present landscape. Moreover, this introduction will set the stage for my investigation into the control of Large Language Model (LLM) outputs, a vastly researched area in the world of NLP.

The early years of NLP were described by rule-based systems, where linguists and computer scientists crafted sets of grammatical rules to parse and generate text. These systems, though labor-intensive and often full of limitations, marked the first steps toward human-computer language interaction.

The 1970s and 1980s witnessed the emergence of statistical approaches to NLP, led by advances in machine learning and data availability. Researchers started to explore methods such as statistical language models, which enabled computers to handle language tasks like speech recognition, sentiment analysis and machine translation by using probability to predict the next word in the sequence given the words that precede it.

The late 20th century and the early 21st century witnessed a big shift in NLP with the introduction of deep learning. Deep neural networks, inspired by the structure of the human brain, revolutionized the field. Using neural networks, like Recurrent Neural Networks [16] (RNNs) and later, Transformer [18] architectures, allowed NLP models to capture the contextual nuances and relationships within a language. One of the groundbreaking milestones during this era was the introduction of Word2Vec [8] in 2013, which enabled the representation of words as dense, continuous-valued vectors. This innovation paved the way for the development of word embeddings,

making it possible to train models on massive text corpora to learn language patterns and semantics. Another big moment breakthrough came with the release of GPT-2 [14] (Generative Pre-trained Transformer 2) model by OpenAI in 2019. GPT-2 demonstrated the remarkable capabilities of large-scale pre-trained language models in generating coherent and contextually relevant text. It marked the transition from rule-based and statistical methods to data-driven, end-to-end neural architectures.

While these advancements have brought NLP to an exciting place, they have also raised concerns regarding the control of language models like GPT-2. With the ability to generate human-like text on a massive scale, there is a pressing need to address issues related to misinformation, bias, and the potential misuse of these models. This fact has introduced a research area where the aim is to maintain control over the generated output of the model. In some cases this comes in a direct form like requiring a token (words, subwords, or characters) to be generated, in other cases it is more shallow, like influencing the sentiment of the generated text or making it sound more human by keeping the output easily readable and understandable.

The stage where we can influence the output of a model directly is in the decoding step, where the to-be-generated tokens are chosen based on a probability distribution. Plenty of strategies are used today, each having advantages over the others. One straightforward approach is greedy decoding. Here, at each step, the model selects the token with the highest probability as the next one. While this is computationally efficient, it often results in suboptimal outcomes as it doesn't consider the broader context. To address the limitations of greedy decoding, beam search is commonly employed. Beam search maintains a fixed number of top-scoring candidates, allowing the model to explore multiple possibilities. It typically yields better results than greedy decoding, but it may still generate repetitive phrases. For introducing randomness into the generated text, top-k sampling is used. At each step, it selects the next token from the top-k most probable tokens, where k is a predefined hyperparameter. This approach provides some diversity while maintaining control.

Throughout my thesis, will explore the methods mentioned above in the hope of guiding the models toward generating desired outputs while minimizing unintended results.

The structure of my thesis is as follows. In Chapter 2 I give an introduction to transformers and NLP's advancement through it's history. I close the chapter with an introduction to decoding strategies and their common types. In Chapter 3 I implement different decoding strategies and compare the most widely used ones on the HunSum-1 model. In Chapter 5, I summarize and conclude my thesis.

Chapter 2

Background

2.1 Transformer Models

In recent years, the field of natural language processing (NLP) has witnessed a paradigm shift with Transformer [18] models. These models have not only revolutionized the way we process and generate human language but have also found applications across various domains, from text summarization to speech recognition and image processing.

The start of the transformer [18] architecture, as introduced by Ashish Vaswani in 2017, marked a significant departure from traditional sequence-to-sequence models, such as recurrent neural networks [16] (RNNs) and convolutional neural networks (CNNs), by introducing a new mechanism for capturing contextual information and dependencies in sequential data.

The core innovation of transformer models lies in their attention mechanism, which allows them to attend to different parts of an input sequence simultaneously. Transformer models have continued to evolve and give rise to numerous variants, such as BERT [3] (Bidirectional Encoder Representations from Transformers), GPT-2 [14] (Generative Pre-trained Transformer), and T5 [15] (Text-to-Text Transfer Transformer), each tailored to specific NLP applications. These models, often pre-trained on large text corpora, have demonstrated remarkable generalization capabilities when fine-tuned, achieving human-level or even superhuman performance in tasks like question-answering, language translation, and sentiment analysis.

In this Chapter, I will guide the reader through different NLP solutions that ended up contributing to the birth of the transformer model. I will be showing the transformer architecture and diving into state-of-the-art decoding strategies.

2.1.1 Neural Networks

2.1.1.1 Neurons

Before explaining a feedforward neural network, it's essential to introduce the fundamental building blocks of neural networks: neurons. Neurons, often referred to as nodes, are the core computational units in a neural network.

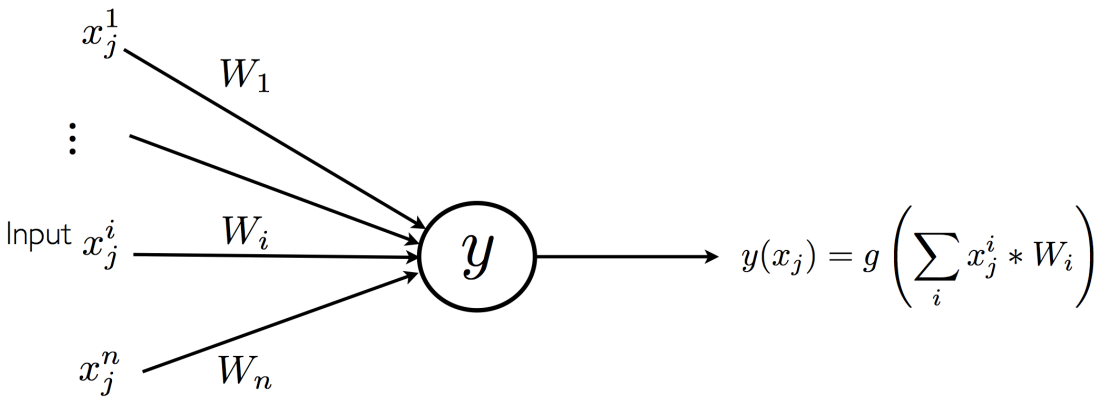

Figure 2.1: Representation of a Neuron.[1]

Each neuron has one or more input variables with weights assigned to them. These weights can be alerted to improve the neuron's results. In order to get an output, we have to sum the input variables multiplied by their weights, this value then goes through an activation function.

$$\text{Output} = \text{Activation}\left(\sum_{i=1}^{n} \text{Weight}_i \times \text{Input}_i\right) \tag{2.1}$$

2.1.1.2 Activation Functions

The most common activation functions used are sigmoid, ReLU, or tanh. Activation functions map input values to specific output ranges, adapting to different data characteristics.

$$f(x) = \max(0, x) \tag{2.2}$$

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.3}$$

[1] https://plato.stanford.edu/entries/artificial-intelligence/neural-nets.html

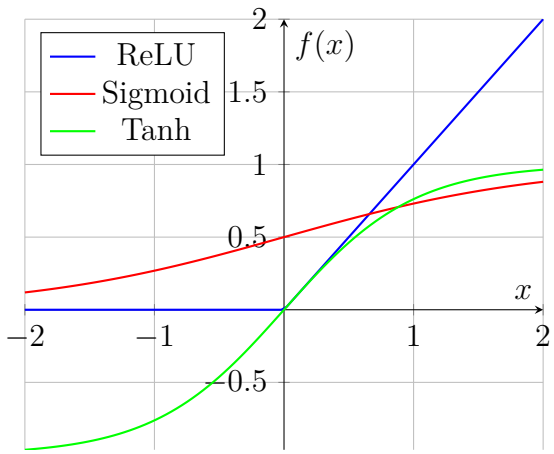

Figure 2.2: Activation Functions: ReLU, Sigmoid, and Tanh.

$$f(x) = \tanh(x) \tag{2.4}$$

2.1.1.3 Feedforward Neural Networks

2.1.1.4 Forward Pass

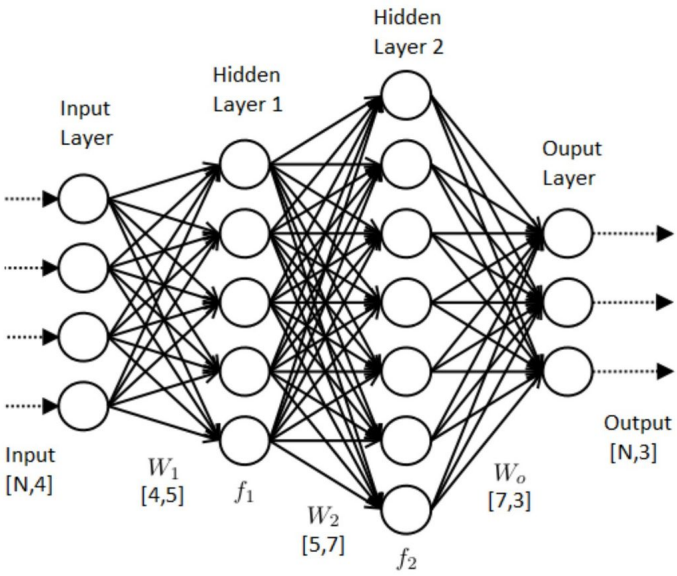

Figure 2.3: Illustration of a Neural Network.[2]

Neurons are organized into layers within a neural network. The network typically consists of an input layer, one or more hidden layers, and an output layer. Information flows from the input layer through the hidden layers to the output layer in

[2] https://www.datasciencecentral.com/the-artificial-neural-networks-handbook-part-1/

a feedforward manner. The connections between neurons, defined by weights, are adjusted during training to optimize the network's performance for specific tasks.

2.1.1.5 Backward Pass

So far I have covered in previous sections the forward pass of the neural network. This step concludes with an output that can be regarded as the output of the network. Loss functions can be utilized at this stage to evaluate the output and measure the error. One of the most widely used loss functions for binary classification is cross-entropy loss.

$$L = -\frac{1}{N_c} \sum_{i=1}^{N_c} y_i \log(\hat{y}_i) \tag{2.5}$$

In order, to improve the model, it is essential to somehow tweak the weights of every neuron. This is why the backward pass exists. The loss of the network's output gets fed backward to adjust the weights of every neuron with the help of a backpropagation algorithm such as gradient descent. Now that we are familiar with both steps of the feedforward neural network, we have a clear understanding of how information flows through the network, from the initial input data in the forward pass to the weight adjustments in the backward pass.

Usually, the input data does not fit into memory so batching has to be implemented. Baching is the process of dividing up our dataset into smaller chunks and feeding these batches to the model. A hyperparameter can be used to choose the size of each batch. One full pass of the dataset in the forward and backward steps is called an epoch. The model can train on multiple epochs but it could lead to overfitting.

The iterative process of forwarding and backward-propagating gradients forms the backbone of training neural networks, enabling them to learn from data and continually improve their performance. By fine-tuning the weights through backpropagation guided by loss functions, neural networks become capable of making accurate predictions and solving a wide range of complex tasks.

2.1.1.6 Hyperparameters

In previous sections, I have already touched on some hyperparameters that are used through the training process, such as batch size, number of epochs, activation functions, loss functions, and optimizers. While getting these parameters right is vital, there are also some more that should be mentioned.

The learning rate is a critical hyperparameter governing the step size at which the model's weights are updated during training.

Early stopping is another valuable hyperparameter that monitors a chosen metric during training and stops the process when the model's performance on a validation dataset starts to degrade. This hyperparameter aids in preventing overfitting and ensures that the model generalizes well to unseen data.

Similarly, the dropout rate is another essential hyperparameter employed for regularization. It determines the probability of dropping neurons during training to reduce the chances of overfitting and improve the model's generalization capabilities.

2.1.2 Recurrent Neural Network

Recurrent Neural Networks [16] (RNNs) are a type of neural network that is specifically designed to recognize patterns in sequential data. This type of data can include text, handwriting, and spoken words. Unlike traditional feedforward neural networks, RNNs have loops that allow them to store information over time, which makes them particularly suitable for tasks that involve sequential or time-series data. In simpler terms, RNNs are great for finding patterns in data that change over time.

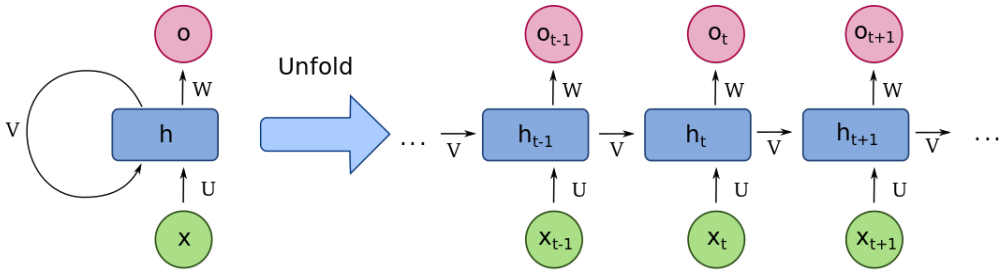

Figure 2.4: Recurrent Neural Network unfolded.[3]

A standard RNN works by unfolding over time or sequence steps, maintaining a hidden state that captures information about a portion of its input. The network performs the same task for every element of a sequence, with the output depending on the previous computations. Mathematically, the hidden state h_t at time t is computed as:

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b_h) \tag{2.6}$$

[3] https://en.m.wikipedia.org/wiki/File:Recurrent_neural_network_unfold.svg

where σ is the activation function, W_{hh} and W_{xh} are weight matrices, b_h is the bias, and x_t is the input at time t. The output at time t is then computed as:

$$y_t = W_{hy}h_t + b_y \tag{2.7}$$

RNNs are used in a variety of applications including natural language processing (NLP), speech recognition, and video analysis. However, standard RNNs suffer from the vanishing or exploding gradient problem which makes it difficult to train them on long sequences. Variants such as Long Short-Term Memory (LSTM) networks were developed to overcome these issues, enabling the learning of long-term dependencies in data.

2.1.3 Long Short-Term Memory

Long Short-Term Memory (LSTM) [5] networks, an extension of traditional Recurrent Neural Networks [16] (RNNs), were introduced to overcome the vanishing gradient problem that plagues RNNs during the training phase. Unlike standard RNNs, LSTMs possess a more intricate cell structure that incorporates three gates: an input gate, a forget gate, and an output gate. These gates, along with a cell state, effectively manage the information flow through the network, Allowing it to keep or remove information over long sequences of data.

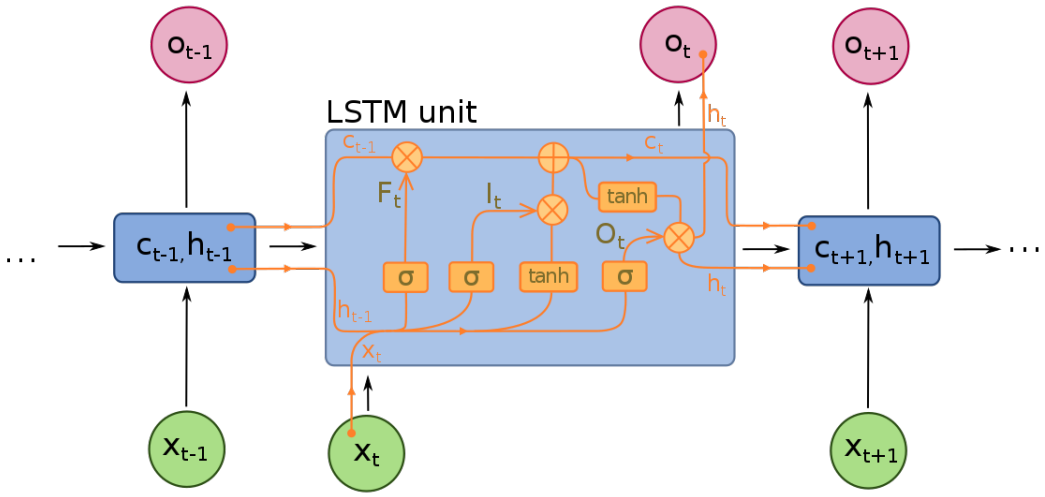

Figure 2.5: One LSTM unit.[4]

LSTMs have deeply impacted machine learning. They have played a pivotal role in enhancing Google's speech recognition, machine translation on Google Translate, and even the responses generated by Amazon's Alexa. Due to their effectiveness in

[4] https://en.m.wikipedia.org/wiki/File:Long_Short-Term_Memory.svg

handling sequential or time-series data, LSTMs are widely used in natural language processing, and speech recognition.

2.1.4 Transformer architecture

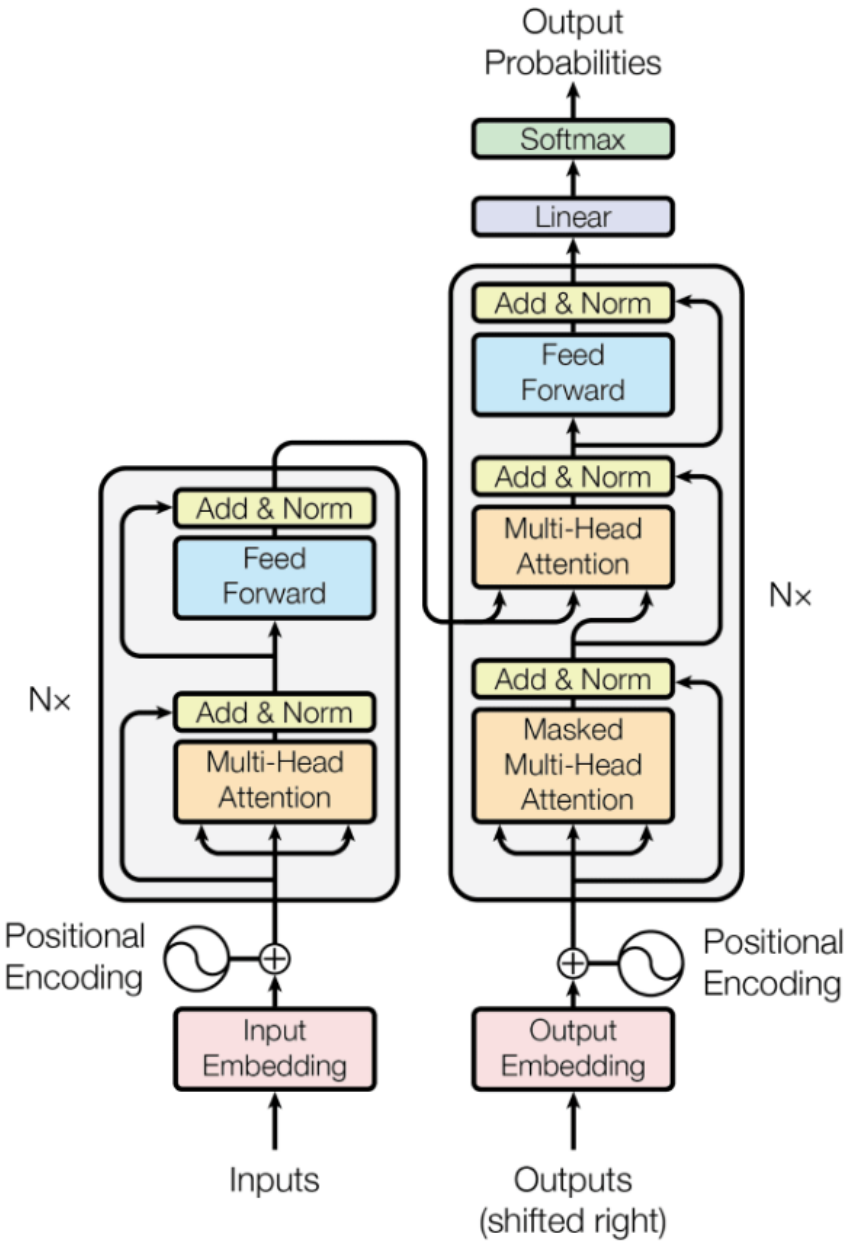

Figure 2.6: The Transformer architecture. [18]

In 2017, the Transformers architecture was introduced by Ashish Vaswani, which took NLP to unprecedented heights. Capable of performing multiple tasks after fine-tuning made transformers the new go-to model in NLP. These models learn through self-supervision, automatically computing an output based on the input. Looking

at the general architecture of Transformers it contains two primary elements called encoders and decoders. The input is received by the encoder, which then constructs its features of it. The decoder uses the encoder's features along with other inputs to generate a target sequence.

The encoding segment comprises six identical, yet independently weighted encoders. Each encoder is split into two parts: a multi-head self-attention mechanism, and a feed-forward neural network. Each sub-layer is enhanced with a residual connection, followed by layer normalization to aid in training stability and model generalization. The final encoder's outputs are channeled to the decoder's Multi-Head Attention layer.

The decoder segment comprises six identical, yet independently weighted decoders. Unlike the encoder with its two sub-layers, the decoder has three, the two found in the encoders and an additional masked multi-head attention layer. Each sub-layer is enhanced with a residual connection, followed by layer normalization, similar to the structure in the encoders.

2.1.4.1 Scaled Dot-Product Attention

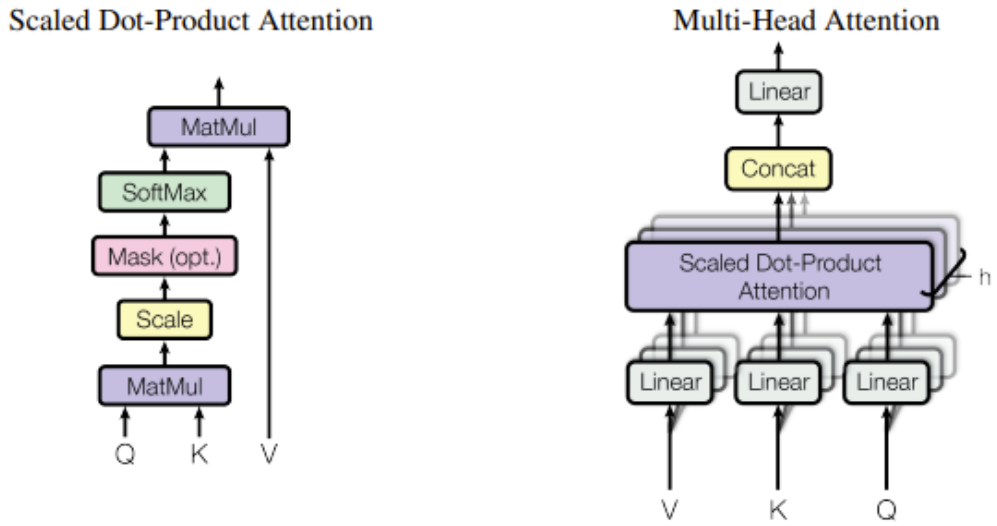

Figure 2.7: The Scaled Dot-Product Attention layer and the Multi-Head Attention layer. [18]

The Scaled Dot-Product Attention layer is for identifying relationships in the input data, no matter where they occur in the input sequence. This attention mechanism computes the attention scores by taking the dot product of the query and key vectors

and then scales down the scores by the square root of the dimension of the key vectors, mathematically represented as:

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

where Q, K, and V denote the query, key, and value matrices respectively, and d_k is the dimension of the key vectors. The scaling factor, $\sqrt{d_k}$, prevents the dot product from growing too large which, could result in vanishing gradients. The softmax function ensures the attention scores are normalized, forming a probability distribution. These normalized scores are used to take a weighted sum of the value vectors, which is the output of the attention layer.

2.1.4.2 Multi-Head Attention

The Multi-Head Attention mechanism is used to allow the model to focus on different parts of the input. The same calculation is done that was mentioned above, but h times. The outputs of these attention heads are concatenated:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \ldots, \text{head}_h)W^O \tag{2.9}$$

where

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{2.10}$$

Here, h is the number of heads, and W^O is an added weight matrix. Each head has its own set of matrix weights associated with them W_i^Q, W_i^K, and W_i^V, which are multiplied by their respective query, key, and value.

2.1.4.3 Position-wise Feed-Forward Networks

Each layer in the encoder and the decoder incorporates a fully connected feed-forward network. This network consists of two linear transformations with a ReLU activation function in between.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2.11}$$

While the transformations stay the same across different positions, the parameters used in each layer differ from each other.

2.1.4.4 Positional Encoding

Positional encoding is required because the model itself does not have any inherent notion of order. Unlike RNNs, Transformers process all positions in the input sequence in parallel which makes them highly efficient, but they don't take into account the order of tokens.

To overcome this, positional encodings are added to the embeddings at the bottoms of the encoder and decoder stacks. These are vectors designed to represent the position of the tokens in the sequence. This way, the model can learn to use the order information to better understand and generate meaningful representations of the sequence data.

2.1.5 BERT

BERT [3] stands for stands for Bidirectional Encoder Representations from Transformers. The model was proposed by Jacob Devlin and is considered the state-of-the-art model for most NLP tasks. It is based on the so far discussed Transformer architecture, but with a novel idea. It is a bidirectional transformer, meaning it is capable of interpreting input sequences not only from the left but the right as well. BERT was pre-trained on a large corpus comprising the Toronto Book Corpus and Wikipedia. Here's a breakdown of the BERT models and their parameters:

- **BERT-Base**:
  - Layers: 12
  - Hidden units: 768
  - Attention Heads: 12
  - Parameters: 110 million
- **BERT-Large**:
  - Layers: 24
  - Hidden units: 1024
  - Attention Heads: 16
  - Parameters: 340 million

2.1.5.1 Input Representation

In BERT, the way data is fed into the model is quite intuitive. Each word or token from the text is represented using a mix of three different kinds of information: a token embedding, a segment embedding, and a positional embedding. The token embedding captures the essence of the word itself, the segment embedding indicates if the word belongs to the first sentence or the second sentence (when comparing two sentences), and the positional embedding tells the model about the position of the word in the sentence.

All three of these pieces of information are combined to create a rich representation of each word. Besides, BERT uses two special tokens, [CLS] at the start and [SEP] to separate sentences, to understand sentence boundaries, and to prepare for tasks like classifying texts.

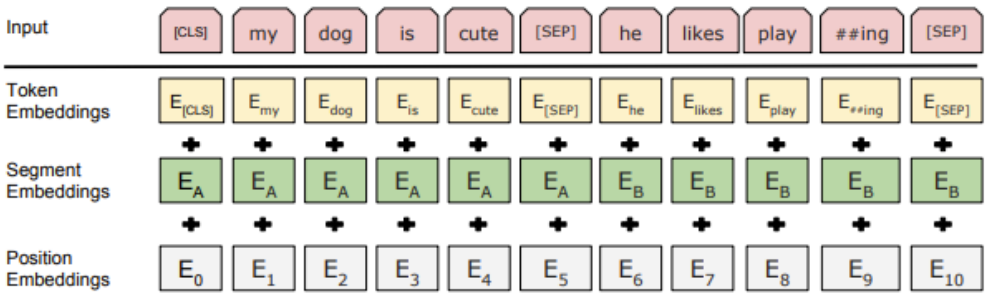

Figure 2.8: BERT input representation. [3]

2.1.6 mBERT

After the initial success of BERT, Google introduced the multilingual version of their large language model called mBERT [13]. This model was trained on a corpus that contained different monolingual Wikipedia pages. There are 104 languages in the corpus including Hungarian, English, and many more. The advantage of these multilingual models is that it is easy to fine-tune them on one specific language making it excel at inputs written in that language.

2.1.7 mT5

The mT5 [15] model is a variant of the T5 (Text-To-Text Transfer Transformer) model, but with a multilingual capacity, making it capable of understanding and generating text in multiple languages, including Hungarian.

The core idea behind mT5 (and T5) is to convert every task into a text-to-text format, meaning that both the input and output are treated as sequences of text. This approach simplifies the process as everything, from translation to classification, is handled as a text rewriting task. The mT5 model is pre-trained on a large corpus of publicly available text from the web, making it a robust foundation for fine-tuning specific tasks or datasets. The simplicity and versatility of the mT5 model, along with its multilingual capabilities, make it a suitable choice for a variety of natural language processing tasks.

2.1.8 HunSum-1

The HunSum-1 [2] is a dataset consisting of 1.14 million Hungarian articles. The purpose of the dataset is to fine-tune abstractive summarization models for the Hungarian language.

Site	**Count**
24.hu	359.4k
origo.hu	305.0k
hvg.hu	216.8k
index.hu	154.5k
nepszava.hu	56.7k
portfolio.hu	22.6k
m4sport.hu	17.9k
metropol.hu	11.1k
telex.hu	4.4k

Table 2.1: Number of data samples per site. [2]

Two language models were fine-tuned with the help of HunSum-1: an encoder-decoder model, where the sides are composed of BERT models, and the weights were initialized with huBERT [9]. The other model is a pre-trained multilingual mT5. These models were used throughout my thesis to test out the different decoding strategies.

2.2 Decoding Strategies

With the rise of the Transformers architecture, the objective to improve every step of the process became highly prioritized. Decoding strategies are one of these domains that are pivotal in natural language processing, particularly when it comes to generating text from language models.

Once a model is trained and enters the inference phase, the decoding strategy dictates how the model constructs textual output sequences, whether that's answering a question, translating text, summarizing the input, or any other text generation task. Different strategies come into play to navigate the balance between computational efficiency and the quality of generated text.

The choice of decoding strategy can significantly impact the coherence, relevance, and diversity of the model's output, so understanding the depths of these strategies is important.

2.2.1 Greedy Decoding

Greedy Decoding is the most straightforward strategy. At each step, the model selects the word with the highest probability as the next word in the sequence and proceeds to the next step. This method is computationally efficient and fast, making it a suitable choice for real-time applications. However, its major downside is that it often produces less coherent or meaningful text as it doesn't consider future implications of the current choice, merely opting for the most probable word at each step. At each time step t, the word w with the highest probability given the previous words is selected:

$$w_t = \arg\max_w P(w|w_1, w_2, ..., w_{t-1}) \tag{2.12}$$

2.2.2 Beam Search

Beam Search is a more refined strategy than Greedy Decoding. Instead of considering only the most probable word at each step, it keeps track of a fixed number of the most probable sequences, called beams, at each step. By considering multiple sequences, Beam Search tends to generate more coherent and contextually relevant text compared to Greedy Decoding. However, it comes at the cost of increased computational complexity as the model needs to maintain and compute probabilities for multiple sequences at each step, making it slower. Beam Search maintains a set of the b most probable sequences (beams) at each time step t:

$$\text{Beams}_t = \text{Top}_b(\{\text{Beams}_{t-1} \times P(w|w_1, w_2, ..., w_{t-1})\}) \tag{2.13}$$

[5] https://huggingface.co/blog/how-to-generate

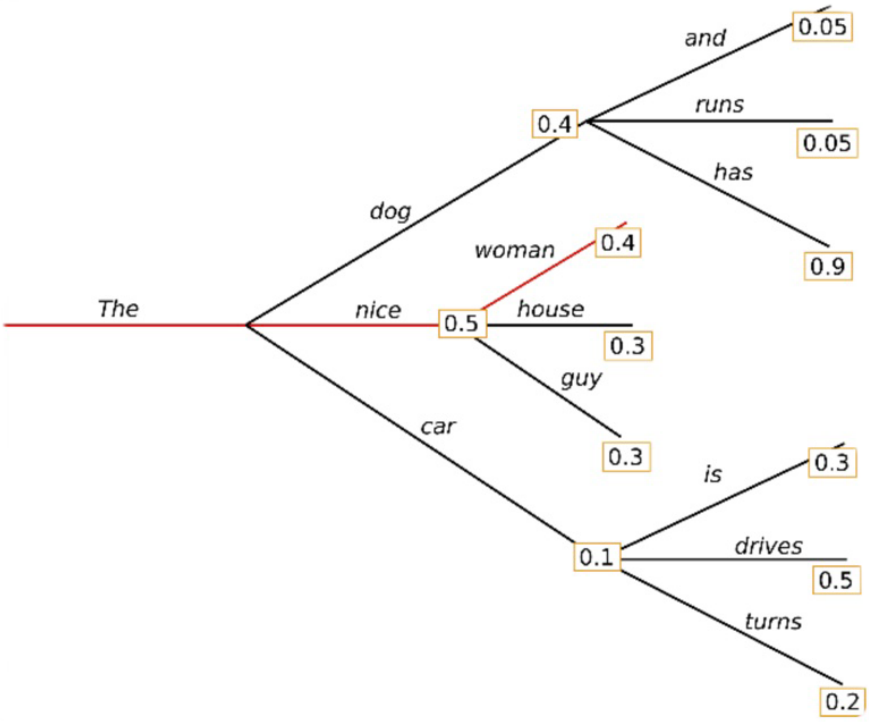

Figure 2.9: Greedy Decoding example. [5]

Example: At $t = 1$, Beam Search keeps track of two of the highest options ($b = 2$), ("The dog") and ("The nice"). At $t = 2$, ("The dog has") sequence has a 0.36 higher probability than ("The nice woman"), which has 0.2. Beam Search ends up choosing ("The dog has").

2.2.3 Diverse Beam Search Decoding

Diverse Beam Search [19] aims to mitigate the lack of diversity in the sequences generated by traditional Beam Search. Instead of only focusing on the highest probability sequences, it introduces a diversity term into the function to encourage the exploration of different paths.

2.2.4 Multinomial Sampling

Multinomial Sampling is another decoding strategy where the next word w is sampled from the entire vocabulary based on the probability distribution over words provided by the model at each time step t:

[6] `https://huggingface.co/blog/how-to-generate`

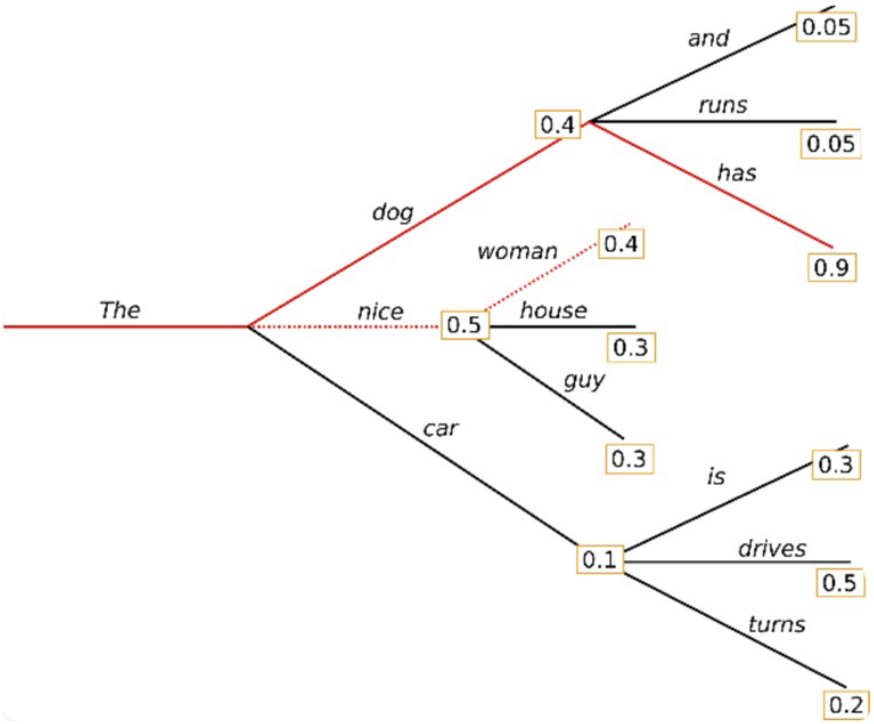

Figure 2.10: Beam Search example. [6]

$$w_t \sim P(w|w_1, w_2, ..., w_{t-1}) \tag{2.14}$$

Multinomial Sampling generates text with a higher degree of randomness compared to Greedy Decoding and Beam Search. This randomness can result in more creative and diverse text generation, but may also produce less coherent or grammatically correct text.

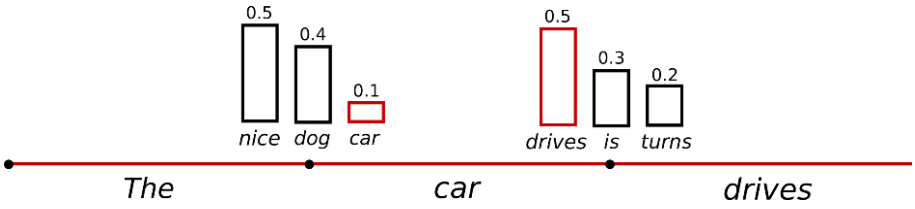

Figure 2.11: Multinomial Sampling example. [7]

[7] https://huggingface.co/blog/how-to-generate

2.2.5 Top-K Sampling

Top-K Sampling [4] introduces randomness in the generation process, which can lead to more creative outputs. At each step, instead of considering all possible next words, it narrows down the choices to the top K most probable words and selects the next word from this group. This method balances the determinism of Greedy Decoding and the computational intensity of Beam Search, offering a middle ground. At each time step t, a subset of K most probable words is selected, and the next word w is sampled from this subset:

$$\text{Subset}_t = \text{Top}_K(P(w|w_1, w_2, ..., w_{t-1})) \tag{2.15}$$

$$w_t \sim \text{Subset}_t \tag{2.16}$$

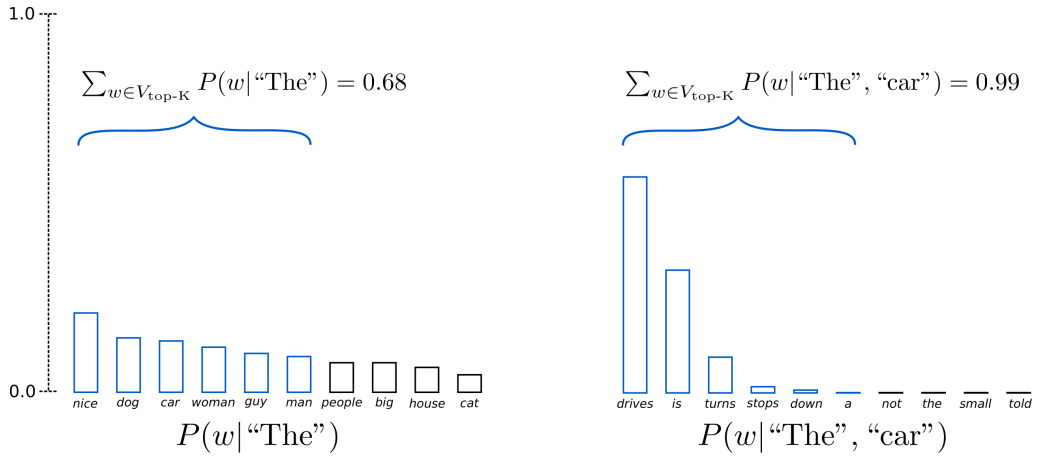

Figure 2.12: Top-K Sampling example. [8]

2.2.6 Top-P (Nucleus) Sampling

Top-P or Nucleus Sampling [6] takes the idea of Top-K Sampling further by dynamically selecting the subset of words to consider based on a cumulative probability threshold, P. Instead of a fixed number of top words, it considers as many top probable words as needed until their cumulative probability exceeds the threshold P. By adjusting the threshold P, one can control the level of diversity in the generated text, making Top-P Sampling a flexible and adaptable decoding strategy. At each time step t, a dynamic subset of words is selected such that the cumulative probability is less than P, and the next word w is sampled from this subset:

[8] https://huggingface.co/blog/how-to-generate

$$\text{Subset}_t = \{w | \sum P(w|w_1, w_2, ..., w_{t-1}) \leq P\} \tag{2.17}$$

$$w_t \sim \text{Subset}_t \tag{2.18}$$

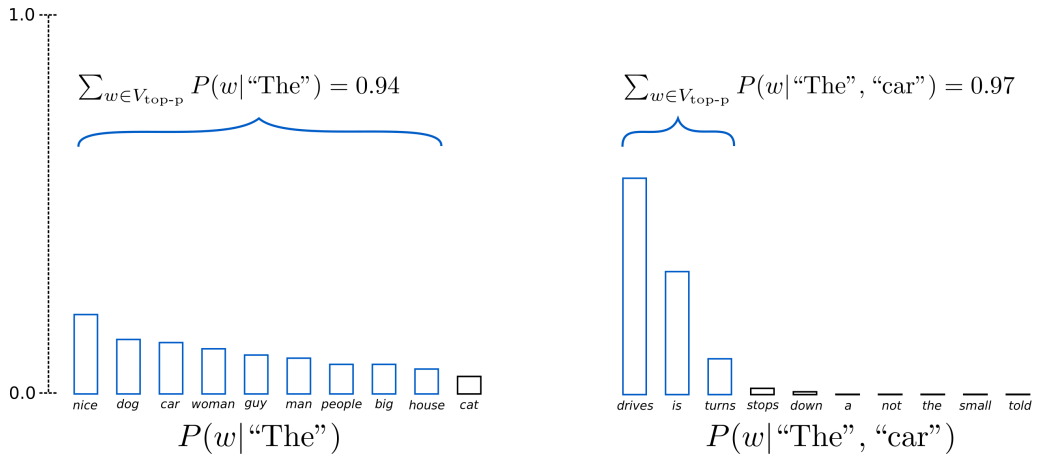

Figure 2.13: Top-P Sampling example. [9]

[9] https://huggingface.co/blog/how-to-generate

Chapter 3

Models

3.1 HunSum-1

Throughout the research process, two models were used to test out the different decoding strategies and constraints. These models were introduced in the HumSum-1 research paper by myself and my fellow research partners.

My main contributions were gathering data for the dataset from different Hungarian news portals. I was involved in the cleaning and deduplication of the dataset and I helped out with the evaluation of the models that were fine-tuned on the HunSum-1 dataset.

The two models that I experimented with are an mT5 Base model and a Bert2Bert model. Both of these were fine-tuned on the HunSum-1 dataset.

3.2 HunSum-2

During the writing of this thesis, there is ongoing research that is about the new version of the HunSum-1 dataset.

The new features that are introduced in HunSum-2 are new Hungarian sites such as kisalfold, delmagyar, and all regional sites. Furthermore, the data-cleaning process was extended.

My contributions were once again about finding new sites that could be integrated into the dataset and parsing these sites. Also did a lot of exploratory data analysis with the goal of finding malicious patterns in our data such as sports results and lottery number announcements.

New models are introduced as well, having the original models fine-tuned on the new dataset and an extractive summarization model is in the works.

Unfortunately, my research began earlier for this thesis than the release of the HunSum-2 paper, so I could not work with these improved models. However, one of my future goals is to compare the results of the two versions and see how they compare.

Chapter 4

Constraints

One of the main aspects of this research was finding different ways to influence the output of the large language models.

One problem with LLMs is that the output cannot be predicted and we cannot control what our model will come up with as an output. Introducing constraints to models can combat these challenges. The hard part is figuring out in what way the models could be influenced and what are the limitations of those constraints.

This chapter focuses on constraints that were introduced during the inference phase of the model. In this phase, the model has concluded training, and the weights are now fixed. The manipulation of the output at this stage is cost-efficient and can be applied to any model. These solutions tend to overcome more exact problems such as omitting or forcing exact tokens in the output.

On the other hand, control during training is a more complex approach that can tweak the weights of the model. This approach is more cost-intensive and once the model learns a style it cannot switch. Also, it cannot be used on other models like the inference phase constraints. One example of this could be controlling the sentiment of the model, by labeling the tokens with a sentiment score and influencing the model to prioritize tokens that have a high sentiment score, meaning they are positive words. This way the model learns that it should be striving towards positive outputs.

Influencing the generation of the model could mean that we drive the model and we could achieve the outputs that we want, but the cost of this approach could be that the outputs become less creative. When the model's generation is tightly controlled or directed, it starts to reflect the biases and limitations of the inputs it is fed, rather than exploring the breadth of its training data.

Moreover, the readability of the outputs could also be affected. While guiding the model helps in generating specific responses, it can also lead to a lack of variety in the language used. The model might start repeating similar phrases, reducing the natural flow and diversity in the language.

It's crucial, therefore, to strike a balance between guiding the model and allowing it enough freedom to utilize its full range of capabilities.

Parameter	**Bert2Bert**	**mT5-base**
`no_repeat_ngram_size`	3	3
`num_beams`	5	5
`early_stopping`	True	True
`encoder_no_repeat_ngram_size`	4	4
`length_penalty`	2	2
`max_length`	128	128

Table 4.1: Parameters of Bert2Bert and mT5 models.

To evaluate the constrained outputs there had to be something to compare it to. I used the original HunSum-1 Bert2Bert and mT5 models as control outputs. All configurations of the constrained and control models were the same except for the constraint.

4.1 Force Token Generation

The first constraint that was implemented and tested out was the most straightforward idea. Giving the option to declare tokens that have to be present inside the generated output of the model.

The user can give multiple strings as input to the generation function, the function then tokenizes these strings and forces the generated tokens to be chosen during the decoding of the output. By including the most important parts of the article in the force token list, this feature can improve the accuracy of the summarization process.

After implementing this constraint into the models, the next task was to test out the advantages and limitations of the solution compared to the original decoding process. For every constraint, I tried to find as many test cases as I could that tested out the feature in different scenarios.

The biggest challenge was that the models tended to generate the force word at the end of the output with no added value to the summary when the presented force word did not fit into the context of the generated text. Another big challenge was trying to find articles that were a good fit for the test case. With this in mind,

there were some cases where the sample size was around 3-4 articles. Since I came up with some niche cases, multiple articles had to be selected from sites that were not present in the original HunSum-1 dataset. This meant that it was a brand new playing field for the model and there was a higher chance for it to be inaccurate since no familiar patterns were present.

The test cases for force token generation were:

Force token already present in the control summary. Generating an output with the control settings of the source text and then choosing one token as a force word. The reason for this test case to exist is to see if the forcing of the word changes anything in the output despite it already being present in the control output.

No difference was identified between the control output and the force token output. Meaning, that the performance of this constraint is identical to the control's if the force word that we want present in the output is already present in the control's output.

Multiple force tokens already present in the control summary. The reasoning behind this test case is the same as the last one, the only thing different is that there are multiple tokens present in the force word list. The goal was to see if the model could handle multiple inputs and with what degree of success.

The output once again turned out to be identical to the control's output meaning that multiple tokens can indeed be added as an input.

Using entities as force tokens. One of the biggest problems of abstractive summarization models, is hallucination. Hallucination refers to a state where the model's output text is incorrect, or not real based on the source text. A lot of the time models tend to hallucinate different nouns into the output making the summary unusable. With this test case, I wanted to reduce the chances of hallucination by forcing some of the nouns present inside the source text to be generated into the output. My focus was specifically on entities, and most importantly, people.

The results are not as clear-cut as before. Most of the time the model can identify where the force token should be placed and steers the output away from hallucination. In some cases, however, the model is unable to inject the force token properly into the output. In these cases either the summary does not make sense or nothing changes from the control version of the output but the force word is put at the very

end of the summary. This to me signals that the model found no logical place for the force word so it just put it at the end, making the summary the least unreadable.

There is a noticeable difference between the two models as well. Bert2Bert seems to do a better job at trying to place the force word in context and mT5 places the force token at the end of the output more frequently.

When dealing with multiple inputs, the models easily get overwhelmed by the number of force words that have to be injected into the summary. This leads to an output where most force words are at the end of the output and some are put into context nicely. If the length constraints are more forgiving the models usually did better with placing multiple force words into a logical context.

Using common words with polysemy as force tokens. Having a word present in the article with polysemy can prove challenging for the model to interpret correctly. For this reason, I tried to find articles where there were ambiguous words in them, added them to the force tokens list, and tested if the output could interpret them properly.

Example

Article: 'Hétfőn a Bors írt arról, hogy Karácsony mindenhová autóval ment a hétvégén, sőt azt írták, hogy a sofőrje még szabálytalanul is parkolt. Emellett a különböző rendezvényeken pont a kerékpározást és a tömegközlekedést népszerűsítette, még villamosokkal is fotózkodott.
A cikk miatt arról kérdeztük a Főpolgármesteri hivatalt, hogy valóban autózott-e, és arról, hogy valóban szabálytalanul parkolt-e.
„A főpolgármester a hétvégén öt különböző fővárosi programon vett részt. A bicikli használatát egy térdsérülés akadályozta. Ellentétben a propagandamédia állításával a szolgálati autó sofőrje nem parkolt a járdán, hanem egy társasház kapubeállójában állt meg a kiszállás idejére szabályosan"
– áll a hivatal válaszában. Arra is rákérdeztünk, hogy a főpolgármester tudott-e volna tömegközlekedési eszközzel menni egyik rendezvényről a másikra, de erre nem kaptunk választ.' [1]
Force token: Karácsony
Control: A Fidesz viszont nem zsarol vagy követel – egyebek mellett erről beszélt a 24.hu-nak adott interjúban Schmitt Pál. A Hit Gyülekezetének vezetője coming outolt.
Constrained: A főpolgármester nem tudta megakadályozni a biciklihasználatot, ezért ment autóval rendezvényekről rendezvényre. A Telexnek adott válaszában úgy nyilatkozott, **Karácsony** nem volt hajlandó tömegközlekedéssel közlekedni, és nem is tudott kerékpározni.

The models had no trouble with using the context of the text as help and placing the force word correctly. Keep in mind that in these cases the force word was already inside the source article. In some cases, the model placed the force word at the end of the output, but no cases were found where the model placed the force word in a different meaning in the output.

[1] `https://telex.hu/belfold/2022/12/13/emberolesi-kiserlet-kes-karacsony-sumeg`

Injecting emotional language into the summary. For this case I wanted to find emotionally loaded articles, name the emotion that was present throughout the article, and try and add that emotion to the force word list. My goal was to try and steer the sentiment of the text in a given direction. It is important to note that the named emotion was not present in the source article, so the model had to find a brand new context for it.

The main problem also arises here, the models tend to generate the force word at the end of the output. In some cases, the force word was generated into the right context and the sentiment was consistent with the force word for that sentence, but the rest of the summary was not influenced by much.

Using idiomatic expressions as force tokens. In the Hungarian language there are a lot of idiomatic expressions that are commonplace to use. Having these understood by the model could result in a more creative response.

Example

Article: 'Narendra Modi indiai miniszterelnök a Nemzetközi Olimpiai Bizottság (NOB) 141. ülésszakának megnyitó ünnepségén Mumbaiban megerősítette országa elkötelezettségét a 2036-os olimpia megrendezése mellett, és kijelentette, hogy "minden követ megmozgatnak" ennek érdekében.
Az olimpiai játékok megrendezése régi vágya Indiáénak, a miniszterelnök bizakodását fejezte ki, hogy a NOB támogatását is elnyeri ezekhez a törekvésekhez.
A KÖVETKEZŐ NYÁRI JÁTÉKOKAT A TERVEK SZERINT 2024-BEN PÁRIZSBAN, MAJD 2028-BAN LOS ANGELESBEN RENDEZIK MEG, A 2032-ES OLIMPIA RENDEZŐ VÁROSÁNAK PEDIG AZ AUSZTRÁLIAI BRISBANE-T VÁLASZTOTTÁK.
A NOB nyilatkozata szerint több érdeklődő is van a későbbi játékokra, és Indiát is nagy esélyesnek látják. A dél-ázsiai országnak tett gesztus, hogy a 2028-as olimpiai játékok programjába beveszi a krikettet, amely különösen Indiában örvend óriási népszerűségnek.' [2]
Force token: minden követ megmozgat
Control: A indiai kormány hivatalosan is bejelentette, hogy India támogatja az olimpiai játékok megpályázását - írja a The Daily Telegraph
Constrained: A indiai kormány bejelentette, hogy India támogatja a 2026 - os olimpiát, és a Nemzetközi Olimpia Bizottság is támogatja az eseményt. A Nemzetközi Olimpiai Iroda közleménye szerint a dél - koreai kormány támogatja Indiát, de nem minden esetben támogatja azt. Az olimpia megrendezését a 2040 - es olimpiára beveszi India, így a kriktik is nagy eséllyel pályáznak a játékok megrendezésére. India a jelek szerint nem áll le, folytatja a felkészülést a következő ötkarikás játékokra. Eközben az indiai olimpiai játékok szervezõi is bejelentették : a NDK - n kívül más országokkal is tárgyal **minden követ megmozgat**

Unfortunately, the models could not identify the meaning behind the expressions and instead tried to take them word for word, resulting in incoherent responses.

Injecting historical context into the summary. Since in the dataset there were no news sites related to historical events, I wanted to try and see how well the models can identify them.

[2] https://www.portfolio.hu/gazdasag/20231014/india-minden-kovet-megmozgat-hogy-ove-legyen-a-2036-os-olimpia-645715

The control models tend to skip or briefly mention the events. However, the generated outputs did a better job of placing the event into the spotlight of the summary, but added context was never present, since the model did not learn about the historical event.

Introducing a related concept. In this test case I modified the source article by removing completely a very common concept - like artificial intelligence from a tech-related article - and had that concept added to the force token list. The objective was to have the models find out about the concept and generate it into the summary in a meaningful way.

Example

Article: 'A ChatGPT-t gyártó OpenAI és a Spotify együttműködéséből született fejlesztéssel a podcasterek automatikusan létrehozhatják a műsoraik idegen nyelvű változatait, ráadásul mindezt a saját hangjukon. A Spotify már partnerségre is lépett néhány ismert amerikai podcasterrel, többek között Dax Sheparddal, Monica Padmannel, Lex Fridmannel, Bill Simmonsszal és Steven Bartlettel, hogy az angol nyelvű adásaikat spanyolra fordítsák le. A következő hetekben a francia és német fordítások bevezetését is tervezik.
A fejlesztés alapja az OpenAI nyílt forráskódú beszédfelismerő rendszere, a Whisper, ami képes az angol beszéd átírására és más nyelvekre fordítására. A Spotify új eszköze azonban túlmutat egy sima fordítóprogramon, ugyanis a fordítás a podcasterek szintetizált hangján szólal meg.
„Azáltal, hogy a Voice Translation a készítő saját hangját adja vissza, a hallgatók világszerte minden eddiginél hitelesebb módon fedezhetik fel és inspirálódhatnak a podcasterektől" – mondta Ziad Sultan, a Spotify személyre szabásért felelős alelnöke.' [3]
Force token: mesterséges intelligencia
Control: Automatikusan lefordíthatja a hangját az amerikai és francia podcastereknek szánt új fejlesztés, amelynek köszönhetően az angol műsorokat más nyelveken is le lehet fordítani.
Constrained: A **mesterséges intelligencia** által fejlesztett Voice Translation segítségével egyszerűen lefordíthatók az adások, és ezzel párhuzamosan a hangjukat is átírhatják.

The constrained models were able to place the related concept into the summary even when the control models did not mention it.

Introducing false information. This test case was different from the others because the goal was to get bad results from the generated output. The force token was always a factually incorrect word, like a year that is almost the same as the one present in the source article.

[3] `https://telex.hu/techtud/2023/09/26/sportify-opeanai-podcast-forditas-hang-klonozas`

Example

Article: 'A ChatGPT-t gyártó OpenAI és a Spotify együttműködéséből született fejlesztéssel a podcasterek automatikusan létrehozhatják a műsoraik idegen nyelvű változatait, ráadásul mindezt a saját hangjukon. A Spotify már partnerségre is lépett néhány ismert amerikai podcasterrel, többek között Dax Sheparddal, Monica Padmannel, Lex Fridmannel, Bill Simmonsszal és Steven Bartlettel, hogy az angol nyelvű adásaikat spanyolra fordítsák le. A következő hetekben a francia és német fordítások bevezetését is tervezik.
A fejlesztés alapja az OpenAI nyílt forráskódú beszédfelismerő rendszere, a Whisper, ami képes az angol beszéd átírására és más nyelvekre fordítására. A Spotify új eszköze azonban túlmutat egy sima fordítóprogramon, ugyanis a fordítás a podcasterek szintetizált hangján szólal meg.
„Azáltal, hogy a Voice Translation a készítő saját hangját adja vissza, a hallgatók világszerte minden eddiginél hitelesebb módon fedezhetik fel és inspirálódhatnak a podcasterektől" – mondta Ziad Sultan, a Spotify személyre szabásért felelős alelnöke.' [4]
Force token: iTunes, Meta
Control: Automatikusan lefordíthatja a hangját az amerikai és francia podcastereknek szánt új fejlesztés, amelynek köszönhetően az angol műsorokat más nyelveken is le lehet fordítani.
Constrained: A chatGPT készítői a hangjukat fordítják át más nyelvre, ezzel pedig az adásokat szinte teljesen más verziók formájában tudják lefordítani. Az OpenAI fejlesztésének köszönhetően azonban már csak úgy tudnak fordítást készíteni, mint egy egyszerű fordítóprogrammal, de az új technológia sokkal hitelesebbé teszi az angol és francia nyelvű műsorok terjedését. A podcasterek most olyan eszközöket fejlesztettek ki, amelyek képesek az átírásra és **Meta iTunes**

The generated outputs were not fooled by this information and injected the force words to the end of the output.

Introducing irrelevant information. Almost the same as the false information test case, but this time the force word was a very out-of-context word.

In the same way as before, the constrained models managed to put the force word at the end of the output.

After analyzing all these test cases it was important to not only evaluate the results by hand but measure them with the help of metrics. From the 10 test cases, I disregarded the last two because those cases were purposely trying to worsen the constrained models. I gathered 15 articles each with a test case in mind and evaluated the ROUGE and the BLEU scores of the control and constrained outputs. The outputs were stemmed with the help of the HuSpacy [10] library to get more accurate results.

	Control				Constrained			
Model	R-1	R-2	R-L	BLEU	R-1	R-2	R-L	BLEU
Bert2Bert	22.25	2.46	16.57	**5.90**	**26.12**	**6.19**	**18.9**	5.01
mT5-base	17.36	0.16	14.73	4.68	20.72	3.46	16.08	5.14

Table 4.2: ROUGE recall and BLEU scores for the Force Token Generation constraint. ROUGE-1, ROUGE-2 and ROUGE-L scores are represented as R-1, R-2, and R-L respectively.

[4] `https://telex.hu/techtud/2023/09/26/sportify-opeanai-podcast-forditas-hang-klonozas`

The results however are of lower quality than the original HunSum-1 paper, this could be because the source media sites are not present in the original dataset. Other than that, the ROUGE recall score favors the constrained versions and the BLEU score is very close but favors the control. Taking into account that with the constrained method we were able to influence the generation in any way we wanted and still got a better performance is a great result. Some negatives however are, the fact that we have to collect force words by hand every time we want to run the constrained version and the fact that sometimes the model disregards the force word and puts it at the end of the output. In Section 4.3, I try to tackle the collection of force words problem.

4.2 Omit Token Generation

Being able to force token generation is a vital feature for summarization, but the inverse of this task is just as important. The idea of omitting tokens declared by the user has some useful use cases. This constraint can overcome the problem of models generating inappropriate sentences, it can be used to try and combat hallucination and it can force the model to make more creative outputs.

The user can give multiple strings as input to the generation function, it tokenizes these and omits every occurrence of the token during decoding.

As with the forcing constraint, declaring test cases and comparing them to a control output seemed like a great way to measure the omit constraint's success. The use cases here are more limited and not as broad.

Here are the test cases for omit token generation:

Choosing one omit token. This test case's purpose was to see if the implemented feature works the way I intended it to. I generated the control output and selected one important string from the summary, placing it into the omit tokens list.

The constrained model decided to ignore generating that word and came up with a very different response that still encapsulated the meaning of the source text.

Choosing multiple omit tokens. Making it work for one word was a great enough achievement, but testing out if the models stay accurate with multiple omitted words was another question to be investigated.

Introducing only a few (3-4) omitted words worked well and the output was still decent. However, the bigger the list of omitted words is the harder it is for the model

to find high-probability sequences in the decoding phase, leading to the response being harder to read.

Using entities as omit tokens. Nouns present inside the summary are usually important parts of the context. I wanted to see how the model would approach the task of not generating the important nouns in the summary but still making it a helpful output.

Most of the time the model tried to be tricky and restructured the output sentences so that it could use an inflected form of the omitted word. Only when I added most inflected forms to the omitted words list was I able to get responses without the force words. In these cases, the model performed worse but the output was still understandable.

After evaluating the results by hand, it was time to see what the metrics tell us. I gathered 15 articles that related to the test cases, then I generated the control of every article, skimmed through the output, and chose words that I thought were not useful in the summary. These words made up the omitted words list and that is how I ran the constrained test cases.

	Control				Constrained			
Model	R-1	R-2	R-L	BLEU	R-1	R-2	R-L	BLEU
Bert2Bert	28.90	**11.11**	18.86	6.16	21.22	3.11	15.43	**6.62**
mT5-base	29.42	8.57	14.71	5.57	**31.50**	2.35	**22.58**	6.52

Table 4.3: ROUGE recall and BLEU scores for the Omit Token Generation constraint. ROUGE-1, ROUGE-2 and ROUGE-L scores are represented as R-1, R-2, and R-L respectively.

Looking at the results we can conclude that the constrained model for mT5-base did the best out of the four. For the Bert2Bert models, the result is not straightforward. The constrained model seems to reach the best BLEU score, but its ROUGE scores lag behind the other models.

Taking into account the fact that the constrained models can be influenced by the user and have a higher chance of generating an output that is to our liking, the results are very positive. The constrained models remained competent with the control models and in some cases even did better.

Using the Omit Token Generation constraint proved to be a useful feature when trying to steer the model's outputs by reducing its list of words to work with. The negative aspect of the constraint is that every force word has to be identified by the user, taking up a lot of time, and the fact that sometimes all the model does is

inflect the omit word making the outputs very similar to the control cases. In later chapters, I try to find solutions to these problems at hand.

4.3 Named Entity Recognition

In Section 4.1 and Section 4.2 the disadvantages of the constraints were identified. Finding a solution to tackle these problems could lead to easier usage and more accurate predictions by the models.

With this in mind, I implemented a novel idea. Using a Named Entity Recognition (NER) model during the inference phase can save time for the user, by not forcing them to find omit or force words by hand.

Named Entity Recognition, also known as NER, involves identifying and classifying key elements in text into predefined categories. These categories typically include names of persons, organizations, locations, quantities, and sometimes more specific items like medical codes. In this constraint, however, only the names of persons were the objective to identify.

The goal of NER is to extract information from large blocks of text. This can be particularly useful in applications, such as the task at hand, summarization.

More advanced NER models use machine learning, particularly deep learning techniques. These models are trained on large datasets, learning to recognize patterns that indicate a named entity. The model used in the inference phase is from the Hugging Face hub. It is a Hungarian Named Entity Recognition model [1] that was fine-tuned on the WikiANN dataset [11].

Introducing an NER model can help to increase the constraints that are in place. This can lead to more accurate and reliable results. The initial idea was to give the source text to the NER model, have it identify every name of persons in the text, and put it into the list of force words. With this version, the user is not required to gather words and the model does the hard lifting.

Not having to declare force tokens by hand during evaluation I was able to run the models on a larger sample size. I selected 100 articles from the HunSum-1 test set and ran the experiment.

Seeing the results was somewhat surprising considering the fact that the hand-selected version did so well against the control models. Observing the metrics, it can be seen, that both constrained models did worse than their counterparts and

	Control				Constrained			
Model	R-1	R-2	R-L	BLEU	R-1	R-2	R-L	BLEU
Bert2Bert	**21.43**	**6.38**	**16.74**	**7.31**	18.30	4.43	13.60	5.53
mT5-base	20.39	5.32	15.68	7.25	17.26	3.88	12.45	5.20

Table 4.4: ROUGE recall and BLEU scores for the Named Entity Recognition constraint. ROUGE-1, ROUGE-2 and ROUGE-L scores are represented as R-1, R-2, and R-L respectively.

the most successful model was the control Bert2Bert for both ROUGE and BLEU scores.

With the results at hand, I suspected that the NER model has faults in identifying names correctly, but looking at some cases by hand, that was not the case. The threshold for selecting names was set to 0.99, meaning the model had to be very confident in an entity to include it in the force words. The problem should be with the generation sequence. In some cases, the names are placed at the end of the output because the model cannot put them in context, resulting in a worse summary. In cases where the model does put the entity in the right context, chances are that the control already did a good job at that task.

As mentioned in Section 4.2, the Omit Token Generation constraint had some disadvantages as well. Choosing tokens to be omitted takes up a lot of time. The idea was to use the NER model to find hallucinated entities and make them part of the omit words list, effectively removing them from the summary. This solution resulted in not having the user search for omitted words and a chance for the Omit Token Generation constraint to reduce hallucination.

This more complex logic required more computational time and did not bring the expected results. The model successfully removed the hallucinated words but added new entities that were hallucinated as well. The function now does three iterations with each iteration searching for hallucinated entities and adding them to the omit words list.

Example

Article: ‘Ma van a trianoni békeszerződés 102. évfordulója, ez a nap 2010. óta Magyarországon a nemzeti összetartozás napja is egyben. Az évfordulóról a pártok és politikusok is megemlékeztek.
Fidesz – A kormánypárt **Orbán Viktor** soraival emlékezett meg az összetartozás napjáról: „Van egy nagy nemzet, amely messze túlnyúlik az országhatárokon. Ennek a lelki közösségnek a megőrzése olyan nehéz száz év után, mint amit megéltünk, mégiscsak sikerült. A nemzeti összetartozás gondolata, érzésvilága erős, talán erősebb, mint a korábbi évtizedekben bármikor is volt.”
KDNP – Közleményükben azt írják, hogy „történelmünk során sokszor váltunk nagyhatalmak játékszerévé, de mindig ragaszkodtunk függetlenségünkhöz, nyelvünkhöz, hitünkhöz, kultúránkhoz, tradícióinkhoz, amelyek biztosították nemzetünk fennmaradását a legnehezebb korszakokban is.” A párt kitért arra is, hogy szerintük a baloldali-liberális politika el akarta magától taszítani a határon túli magyarságot. Továbbá kiemelik, hogy Magyarország segítségnyújtását Kárpátaljának: „Az anyaország mindig segítő kezet nyújt a bajbajutottaknak, most különösképpen a kárpátaljai testvéreinknek”
Karácsony Gergely – „Trianon közös tragédiája a magyar nemzetnek, kitörölhetetlen. Számomra elsősorban édesanyám története, aki a második bécsi döntéssel visszacsatolt területen született” – írja le személyes történetét a főpolgármester. Felhívta a figyelmet, hogy Trianon kérdésében nincsenek oldalak, csak közös gyász. „Jó volna, hogy ha végre megtanulnánk a ”vagy„ helyett ”és„-t használni, és a nemzeti összetartozás napján megfogadni, Trianonból tanulva, hogy magyarokként egyek vagyunk.”’ [5]
Initially Found Entities: Karácsony Gergely, Orbán Viktor
Hallucinated Entites after 1st Iteration: Orbán Viktort, Kocsis Máté, Szatmáry Kristóf
Hallucinated Entites after 2nd Iteration: Bobrovniczky Kristóf, Kókák Imre, Orbán Viktort, Kocsis Máté, Szatmáry Bianka, Szalay, Szatmáry Kristóf
Hallucinated Entites after 3rd Iteration: Bobrovniczky Kristóf, Kocsis Máté, Szatmáry Bianka, Szalay, Kókák Imre, Orbán Viktort, Molnár Zsolt, Szatmáry Kristóf
Output: A Orbán – Trianon megemlékezésen a Fidesz és a KDNP is megemlékezett. A Fidesz **Orbán Viktor** sorával emlékezett, a kormánypártok és az MSZP is. Videó! Videók! A képre kattintva galéria nyílik! FOTÓK : Molnár Ádám / Népszava. FNDATE : FACSONS / MOLNU / SZAVA, MOLSZF / FISZ / RNSA / SONON / FARARSS, SZEGOM, CS CSUU, KNNA / CSOMU.. VMSZ. SZÓS : SZNSS. **Karácsony Gergely**

The results, however, remain the same, the model keeps insisting on generating false entities into the output.

	Control				Constrained			
Model	R-1	R-2	R-L	BLEU	R-1	R-2	R-L	BLEU
Bert2Bert	**21.43**	**6.38**	**16.74**	**7.31**	17.99	4.91	14.07	5.18
mT5-base	20.39	5.32	15.68	7.25	17.13	4.38	12.41	5.22

Table 4.5: ROUGE recall and BLEU scores for the Named Entity Recognition Complex constraint. ROUGE-1, ROUGE-2 and ROUGE-L scores are represented as R-1, R-2, and R-L respectively.

Looking at the metrics for the same samples but with the now more complex logic, it can be identified that no significant changes were achieved. Both the Bert2Bert and mT5-base models did very similarly to the simple case.

Seems like the generation sequence is most effective when hand-picking out words. Using the NER model increased the computational time but achieved less impressive results. The same can be said for the more complex version of the constraint that

[5] `https://telex.hu/belfold/2022/06/04/trianon-102-evfordulo-nyilatkozatok-kdnp-karacsony`

tried to tackle hallucination but failed to do so. This constraint is useful for running manipulation on big samples of data, but a performance drop should be expected from the models.

4.4 Lemmatization of Omit Tokens

During the testing of the Omit Token Generation constraint, I identified a pattern. The model was striving to generate the omitted word with a different inflection.

The problem with this is that the whole reason for the constraint to be introduced is to remove the world from the output whether it is in an inflected form or not. This is especially important for Hungarian where the structure of the language is much more complex than English and inflections are very common.

In order to prevent the model from using inflected forms, lemmatization can be introduced. Lemmatization is the process of reducing words to their base or dictionary form, known as a lemma. Unlike stemming, which crudely chops off the ends of words, lemmatization considers the context and converts the word to its meaningful base form. For instance, the words 'running', 'ran', and 'runs' are all lemmatized to 'run'.

With the help of a lemmatizer model, the words from the summary and the omitted words can be lemmatized. Now that every word is in its base form it is easy to check if the summary contains one of the committed words. If that is the case the original word that is represented by the lemma can be added to the list of omitted words and with the new list a new generation can take place. This logic allows the model to always identify the inflected forms in the summaries and avoid generating them.

Here is a great example with a case where the model does not look for lemmas and a case where lemmatization is introduced.

Example

Inputs:

- **Prompt:** 'Novembertől a nagyközönség számára is elérhetővé vált a Credential Manager nevezetű funkció, amelynek köszönhetően a korábbinál sokkal **biztonságosabb** platformmá válhat az Android – legalábbis a Google szerint. Ahogy arról korábban már az Index is beszámolt, az idén októberben megtartott **Google** I/O-n rengeteg mindenről szó esett. A **Google** amellett, hogy bemutatta legújabb telefonjait, a **Google** Pixel 8-at és a Pixel 8 Prót, illetve új okosóráját, a **Google** Pixel Watch 2-t, egyéb témákra is kitért – például a **biztonságra**.' [6]
- **Initial Omit Words:** Google, biztonságosabbá

Control Output:

- A **Google** szerint az Android **biztonságosabbá** válhat, mint az eddigi platformok.

1st Iteration of Lemmatization:

- Többek között arról is beszélt a keresőóriás, milyen biztonsági funkciót kapott az Android, és hogy mennyire **biztonságos** az operációs rendszer.
- **New Omit Words:** Google, biztonságosabbá, biztonságos

2nd Iteration of Lemmatization:

- Az Android fejlesztői a közösségi oldalukon tették közzé, hogy milyen funkciók vannak az Android operációs rendszerben.

This example showcases that most of the time the model tries to use inflected forms for generation and how effectively the lemmatizer identifies these, resulting in an output that is clean of any omitted words and their inflected forms. Something to pay attention to is the word **biztonsági**. The lemma of this word is biztonság and the lemma of the omitted word is biztonságos. That is why the model did not include it in the list of omitted words.

4.5 Other Ideas

Throughout my research, there were other smaller ideas that I tested out, but I felt like they did not deserve their own section. So now I will briefly summarize these findings in smaller subsections.

4.5.1 Length Constraint

One of the most commonly used constraints in NLP is the length constraint alongside with repetition constraint. These are so essential for achieving decent results, that they are almost always present in the generation stage.

[6] `https://index.hu/techtud/2023/11/05/android-14-credential-manager-jelszokezelo-jelkulcs/`

Take a look at Table 4.1 which shows the parameters used by the models. Several parameters such as `early_stopping`, `length_penalty`, and `max_length` adjust the length of the output. Here are the used parameters for the constrained model:

- `early_stopping`: True
- `length_penalty`: 2
- `max_length`: 128

Despite how commonly these parameters are used I still wanted to measure just how important these parameters are when generating outputs. The sample size of the tests is 100 articles randomly chosen from the test set of HunSum-1.

	Control				Constrained			
Model	R-1	R-2	R-L	BLEU	R-1	R-2	R-L	BLEU
Bert2Bert	19.08	5.57	15.75	5.55	22.80	6.28	17.95	7.63
mT5-base	18.66	6.47	16.92	4.57	**24.44**	**7.02**	**18.39**	**8.20**

Table 4.6: ROUGE recall and BLEU scores for Length constraints. ROUGE-1, ROUGE-2 and ROUGE-L scores are represented as R-1, R-2, and R-L respectively.

The difference in this evaluation from the rest is the fact that the control model does not have any parameter set that influences the length of the output, while the constrained model has every parameter set that controls the length. The results shown are indicative of how important it is to set these parameters.

4.5.2 Repetiton Constraint

The ability to control how often the model repeats itself is vital since without these constraints outputs have a chance to repeat themselves.

Once again taking a look at Table 4.1, it is shown that these constraints were already utilized during the testing of the other constraints. All parameters such as `no_repeat_ngram_size` and `encoder_no_repeat_ngram_size` influence the repetition of tokens. Here are the used parameters for the constrained model:

- `no_repeat_ngram_size`: 4
- `encoder_no_repeat_ngram_size`: 3

	Control				Constrained			
Model	R-1	R-2	R-L	BLEU	R-1	R-2	R-L	BLEU
Bert2Bert	5.09	4.83	4.11	2.21	22.50	6.25	17.27	7.98
mT5-base	**25.28**	**11.45**	**20.53**	**12.82**	20.54	5.19	15.90	7.37

Table 4.7: ROUGE recall and BLEU scores for Repetition constraints. ROUGE-1, ROUGE-2 and ROUGE-L scores are represented as R-1, R-2, and R-L respectively.

The sample size of the tests is 100 articles randomly chosen from the test set of HunSum-1.

The control models did better in both ROUGE and BLEU scores for the mT5-Base model, indicating that setting the repetition parameters is not trivial and should be tested. In the case of Bert2Bert, it is clearly visible that the constrained model beats out the control model, leading me to believe that Bert2Bert is more exposed to repetition. Even though the mT5-Base control model did better, these parameters are still useful for avoiding repetition and should always be considered.

Chapter 5

Decoding Strategies

Other than constraints, the other topic I researched throughout my thesis was decoding strategies. Decoding strategies are used during inference and it is essentially an algorithm that determines what tokens should be selected based on the current array of tokens that the model has already selected. There are a lot of questions involved when we are trying to find the right decoding strategy for our model. We have to look at the quality of the output, diversity, and computational overhead all together to be able to choose the right strategy.

In Section 2.2 I already went over some of the most popular decoding strategies and their inner workings. I will be further exploring these strategies and will be evaluating them on the HunSum-1 test set to compare them.

5.1 Greedy Decoding

This approach is regarded as the most straightforward way to do decoding. As discussed previously, this method simply selects the highest probability tokens. This is a very efficient method in terms of computing and simplicity, but it lacks diversity most of the time leading to shallow, boring outputs.

Model	R-1	R-2	R-L	BLEU
Bert2Bert	**20.28**	**5.85**	**16.29**	**7.22**
mT5-base	8.42	0.75	7.38	4.16

Table 5.1: Evaluation of the Greedy Decoding strategy used by the Bert2Bert and mT5-base models.

Upon evaluating the model on the HunSum-1 test cases, the Bert2Bert model outperformed mT5-base by a significant margin.

5.2 Beam Search

Beam search is the most widely used decoding method due to the fact that it does decent in every metric (computational cost, diversity, output quality). These metrics can change depending on the parameters defined for the search, if selecting a lot of beams the computational overhead can increase significantly, but the diversity increases also. I ran the tests with `num_beams` = 5, meaning that beam search takes into account the 5 most likely tokens. The sequence that has the highest score overall is selected as the output.

Model	R-1	R-2	R-L	BLEU
Bert2Bert	20.64	5.28	16.01	6.62
mT5-base	**21.72**	**5.77**	**16.77**	**7.28**

Table 5.2: Evaluation of the Beam Search used by the Bert2Bert and mT5-base models.

The overall performance of the Beam Search method is superior to the greedy decoding. Between the two models, the mT5-base seems to utilize this strategy to the fullest.

5.3 Multinomial Sampling

Looking at multinomial sampling, this decoding strategy's goal is to create very diverse outputs. That is why at every time step the next token is chosen from a probability distribution of the entire vocabulary.

Model	R-1	R-2	R-L	BLEU
Bert2Bert	16.01	2.86	12.25	5.35
mT5-base	**17.37**	**3.49**	**13.39**	**5.98**

Table 5.3: Evaluation of the Multinomial Sampling strategy used by the Bert2Bert and mT5-base models.

The results are lower than the previous strategies. This could be mainly due to the fact that this method strives for unique outputs and the ROUGE and BLEU scores reward similarities, not uniqueness.

5.4 Multinomial Beam Search

Combining the previous two strategies into one we get Multinomial Beam Search. This strategy is very similar to beam search but deterministically keeping the top N candidates, the algorithm samples from the top N candidates based on their probability distribution. This little change in the beam search algorithm introduces more randomness leading to more unique outputs.

Model	R-1	R-2	R-L	BLEU
Bert2Bert	20.24	5.10	15.68	6.50
mT5-base	**21.52**	**5.83**	**16.83**	**7.30**

Table 5.4: Evaluation of the Multinomial Beam Search used by the Bert2Bert and mT5-base models.

Despite having the goal of generating unique outputs this decoding strategy does well in terms of metrics as well, although, not quite the level of normal beam search.

5.5 Diverse Beam Search

Another version of beam search that once again tries to introduce more diversity to the algorithm is Diverse Beam Search [19]. In Diverse Beam Search, the total number of beams is divided into groups. Each group does its beam search independently. The key feature of Diverse Beam Search is a diversity penalty that scores candidates in each group. This penalty is applied to discourage the selection of candidates who are too similar to those chosen in other groups.

Model	R-1	R-2	R-L	BLEU
Bert2Bert	19.93	5.07	15.62	6.79
mT5-base	**20.85**	**5.37**	**15.98**	**7.17**

Table 5.5: Evaluation of the Diverse Beam Search used by the Bert2Bert and mT5-base models.

Comparing the two modifications of the beam search algorithm it seems like the multinominal sampling infused beam search does better on our test models.

5.6 Top-K Search

Top-K Search is a stochastic method that once again introduces randomness into the algorithm. The way it does it is by selecting the top K words and sampling one from that set. I selected $K = 50$ for my runs.

Model	R-1	R-2	R-L	BLEU
Bert2Bert	16.29	2.88	12.41	5.40
mT5-base	**17.24**	**3.35**	**13.09**	**5.87**

Table 5.6: Evaluation of the Top-K Search used by the Bert2Bert and mT5-base models.

It appears that a new trend is emerging. Most of the time mT5-base slightly outperforms the Bert2Bert model and methods that are striving for more creative outputs lose points on the ROUGE and BLEU scores. This effect is visible in the Top-K method's metrics as well.

5.7 Top-P Search

Top-P Search, also known as Nucleus Sampling is an advancement of the Top-K Search approach. Instead of the top K probable words, this method uses a P value which is an accumulated probability threshold. The list of potential next tokens consists of words that have an equal or higher probability than the threshold. During my runs, I set $P = 0.92$.

Model	R-1	R-2	R-L	BLEU
Bert2Bert	14.50	2.40	11.27	5.11
mT5-base	**17.36**	**3.43**	**13.40**	**5.94**

Table 5.7: Evaluation of the Top-P Search used by the Bert2Bert and mT5-base models.

The results seem to be mostly on par with the Top-K Search strategy when looking at the mT5-Base model. However, the Bert2Bert model did a little bit worse than its counterpart.

5.8 Temperature Sampling

Temperature Sampling introduces a new hyperparameter for the decoding strategy. By adjusting the temperature, T hyperparameter we can influence how diverse the

output should be. Choosing a temperature around the value 0 makes this decoding strategy equal to greedy decoding, meaning that the most probable words are selected. If we end up choosing a higher value, the probabilities shift and words that had a lower probability get a higher chance to be selected. So by setting the temperature parameter high, the output becomes more diverse. Here is the equation for how temperature scaling is implemented in the SoftMax function:

$$P(\mathbf{w})_i = \frac{e^{w_i/T}}{\sum_{j=1}^{K} e^{w_j/T}} \quad \text{for } i = 1, \ldots, K \tag{5.1}$$

Model	R-1	R-2	R-L	BLEU
Bert2Bert	17.41	3.54	13.42	5.80
mT5-base	**19.08**	**4.34**	**14.65**	**6.53**

Table 5.8: Evaluation of the Temperature Sampling used by the Bert2Bert and mT5-base models.

For testing I set $T = 0.7$, meaning that a more diverse output was the goal. The results are continuing the trend, mT5-Base outshining the Bert2Bert model, but the overall performance is lacking behind some of the other decoding strategies.

5.9 Contrastive Search

Contrastive Search [17] uses the Top-K candidates to choose one as the next token. The next token is evaluated by an algorithm that has two main aspects. The first one is the model confidence, meaning the probability of the current word given the current context. The other aspect is the degeneration penalty. It is the maximum cosine similarity between the token representation of the current word and the current context. Looking at these terms the candidate with the highest score is chosen as the next token.

Model	R-1	R-2	R-L	BLEU
Bert2Bert	19.07	4.27	14.71	6.44
mT5-base	**19.73**	**4.62**	**15.55**	**6.74**

Table 5.9: Evaluation of the Contrastive Search used by the Bert2Bert and mT5-base models.

Contrastive Search seems to work equally on both models with mT5-Base being a bit better. Both model's performance is average compared to the other decoding strategies.

Chapter 6

Conclusion

Looking back at my thorough research on controllable text generation I feel satisfied with the results.

I was able to showcase the potential of our HunSum-1 dataset and the models that were fine-tuned on it.

I showcased multiple ways Large Language Models can be controlled to the benefit of the users. From introducing force tokens and omit tokens, to taking these ideas further and making novel algorithms. Despite influencing the natural flow of the output generation these algorithms often performed on the same level as the control models or even outperformed them in some cases. This is a noteworthy achievement since it allows users not only to guide the flow of the generation but also to maintain the level of the control cases.

Furthermore, I researched different decoding strategies, comparing them and seeing how they handle abstractive summarization outputs. The results showed us that beam search does best in terms of metrics, but other methods are worth exploring when looking for more diverse or computationally more efficient solutions.

This field of Natural Language Processing is still in its early stages, and a lot more research could be done on this topic. For future improvements I am planning on testing everything on the HunSum-2 models, I want to create more intricate constraints and fine-tune the hyperparameters of current ones. I also want to explore more niche decoding strategies and do a human evaluation on all of the ones covered in this thesis.

Acknowledgements

I extend my gratitude to my advisor, Judit Ács, for her guidance and insight throughout my research. Her expertise has been a big driving force behind this work.

Bibliography

[1] Taner Akdeniz. BERT-Base Hungarian-Cased-NER Model. `https://huggingface.co/akdeniz27/bert-base-hungarian-cased-ner`. Accessed: 2023-11-29.

[2] Botond Barta, Dorina Lakatos, Attila Nagy, Milán Konor Nyist, and Judit Ács. HunSum-1: an Abstractive Summarization Dataset for Hungarian. In *XIX. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2023)*, pages 231–243, Szeged, Magyarország, 2023. Szegedi Tudományegyetem, Informatikai Intézet.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[6] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

[7] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[8] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[9] Dávid Márk Nemeskey. *Natural Language Processing Methods for Language Modeling*. PhD thesis, Eötvös Loránd University, 2020.

[10] György Orosz, Zsolt Szántó, Péter Berkecz, Gergő Szabó, and Richárd Farkas. Huspacy: an industrial-strength hungarian natural language processing toolkit, 2022.

[11] Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, 2017.

[12] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[13] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*, 2019.

[14] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[15] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[16] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[17] Yixuan Su, Tian Lan, Yan Wang, Dani Yogatama, Lingpeng Kong, and Nigel Collier. A contrastive framework for neural text generation. *Advances in Neural Information Processing Systems*, 35:21548–21561, 2022.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[19] Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016.

Appendix

In these sections, I showcase more examples that I used during my research process but they were not as interesting as the ones that made it to the main part of the thesis. Still, these examples provide more insight into how the constraints perform on different sources.

A.1 Force Token Generation

Take a look at the Force Token Generation constraint. I defined a lot of test cases but only showed a limited amount of examples. Here are some more:

Example

Article: 'Bloys bocsánatkérésében arról beszélt, hogy bár tudja, „hülye ötlet volt", az motiválta, hogy nagyon szenvedélyesen szereti a csatornája műsorait, illetve „azokat az embereket, akik dolgoznak rajtuk". A vezérigazgató arról beszélt, csak azt akarja, hogy „a műsorok nagyszerűek legyenek, az emberek szeressék őket". „Nagyon fontos számomra, hogy mit gondolnak a műsorokról. Erre gondoltam akkor is, amikor 2020–2021-ben otthonról dolgoztam és egészségtelenül sokat lapozgattam a Twittert. Aztán támadt egy hülye ötletem, hogy kiadjam a frusztrációmat" – összegezte Bloys. Bocsánatot kért a kritikusoktól, majd beszélt még arról is, hogy hat fiókot kezelt másfél éven át. Bloys bocsánatkérése egy nappal azután hangzott el, hogy a Rolling Stone közzétett egy cikket, amely részletezi az ügyvezető és az HBO elleni indított pert – ezt a vállalat egy korábbi alkalmazottja, Sumy Temori indította el, aki úgy érzi, jogtalanul rúgták ki őt. A lap ehhez vett elő 2020-as és 2021-es üzeneteket, amelyekben Bloys többször is beszélt arról, hogy kamu Twitter-fiókokkal kéne válaszolni a negatívan író kritikusoknak. Temori akkoriban ügyvezetői asszisztens volt, állította, hogy utasították ezeknek a profiloknak az elkészítésére.' [1]
Force token: HBO
Control: Őt követi Szél Bernadett és Orbán Viktor a Publicus felmérésében.
Constrained: Az HBO vezérigazgatójának és elnökének azért kellett elnézést kérnie, mert azt mondta, nem akarta, ha negatív kommenteket írnának róluk.

Taking a look at the control output, we can determine that it is completely unrelated to the article. The model gives this exact output pretty frequently for some reason. However, adding a force token fixes the output for us.

[1] `https://telex.hu/after/2023/11/03/hbo-vezerigazgato-kamu-profil`

Example

Article: ‘Orbán Viktor az EU 27 tagállamának vezetőit tömörítő brüsszeli csúcstalálkozóra érkezve beszélt erről. A magyar kormánynak nagyon világos és átlátható a stratégiája, ami eltér a többség, „valószínűleg az önök stratégiájától is” – mondta a miniszterelnök egy újságírói kérdésre. „Mindent szeretnénk megtenni a béke érdekében. Ezért minden kommunikációs vonalat nyitva tartunk az oroszok felé, különben nem lesz esély a békére. Ez egy stratégia, ezért büszkék vagyunk rá.
Mi vagyunk az egyetlenek, akik a béke nevében és érdekében beszélünk, ami mindenkinek érdeke lenne Európában”.
Költségvetési emelés: hiányoznak a „politikai érvek” A magyar miniszterelnököt megkérdezték róla, mi a véleménye a többéves uniós költségvetés emeléséről. (Orbán Viktor korábban ezt elfogadhatatlannak nevezte, ebben a cikkünkben írtunk róla, mire kér az EU több pénzt és hol sántítanak a magyar kormány ellenérvei.) „Több pénzt akarnak beszedni a tagállamoktól, köztük tőlünk is, hogy odaadhassák migráció... befogadáspolitikára” és „az ukránoknak” – mondta a miniszterelnök. Az Európai Bizottság ugyanolyan, a migrációt visszafogó megállapodásokra kért többletforrásokat, mint amilyet Tunéziával kötött és Egyiptommal tervez, de Orbán Viktor arról beszélt, „nem akarunk pénzt adni a migránsoknak”.’ [2]
Control: A Viktor - Putyin csúcson a magyar miniszterelnök azt mondta, az EU - nak nem szabad beleavatkoznia az ukrajnai konfliktusba, mert az a béke mellett szól.
Force token: Vlagyimir, Putyin
Constrained: A Viktor magyar kormányfővel tartott csúcstalálkozó után a magyar miniszterelnök azt mondta, hogy nem akar pénzt adni az EU - nak, de azt is elmondta, mit akar az Európai Bizottság, és mire kér pénzt a magyaroktól. A migrációt viszont nem engedjük meg, mert nem akarunk semmit adni a migránspolitikának. Az EU 27 tagországának vezetõit tömörítõ brüsszeli csúcsértekezleten Orbán Viktor kijelentette : büszke az orosz elnökre, akivel az ukrajnai háború miatt találkozott, Vlagyimir Putyinnal pedig a béke mellett kardoskodott.
Force token: Vlagyimir Putyin
Constrained: A Viktor - Vlagyimir Putyin csúcstalálkozón a magyar miniszterelnök azt mondta, az EU - nak nem szabad beleavatkoznia az ukrajnai konfliktusba, mert az a béke és a biztonság politikája. Az EU 27 tagországának vezetõit tömörítõ Európai Unió 27 tagállamainak vezetőit kérdeztük.

Declaring force tokens separately or as one string makes a difference in the output. This example shows one version where ‘Vlagyimir Putyin’ is the force token and one where there are two tokens ‘Vlagyimir’ and ‘Putyin’. I was not able to solve which version does better, but if the user wants to make sure the two words are after each other then it should be passed in as one token.

[2] `https://telex.hu/belfold/2023/10/26/orban-viktor-putyin-buszkeseg-talalkozas`

Example

Article: ‘A vádirat szerint a két barát rendszeresen együtt ivott a helyi kocsmában, 2021. december 24-én viszont nem oda, hanem egyikük házába mentek, ahol iszogattak, zenét hallgattak és beszélgettek. Este tíz óra körül már mindketten ittasak voltak, amikor az egyik férfi kiment a fürdőszobába, majd visszafelé a konyhaasztalról felvett egy 13 centiméter pengehosszúságú kést, a szobába érve pedig minden előzetes ok nélkül mellkason szúrta a barátját, akinek mindössze annyit mondott, hogy
„Ne félj, nem fog fájni!”
A barát a szúrás erejétől a mögötte lévő ágyra esett, és eszméletét vesztette, a férfi pedig távozott.
Amikor a mellkason szúrt férfi magához tért, megpróbálta felhívni a testvérét, aki nem vette fel. Mivel állapota rosszabbodott, éjjel két óra körül egy videót küldött magáról a testvérének, és megírta neki, hogy megkéselték. A testvér ezt már észlelte, értesítette a rendőrséget és a mentőket, akik az életveszélyes állapotban lévő sértettet kórházba szállították. A férfi életét végül a szakszerű orvosi beavatkozás mentette meg.
Az ügyészség a vádiratában az elkövetővel szemben börtönbüntetés kiszabását és a közügyek gyakorlásától történő eltiltását indítványozta azzal, hogy az elkészítő ülésen történő beismerése esetén a Veszprémi Törvényszék kilenc év börtönbüntetésre és kilenc év közügyek gyakorlásától eltiltásra ítélje. Az ügyészség annak megállapítását is indítványozta, hogy bűnössége esetén a vádlott a törvény rendelkezésénél fogva nem bocsátható feltételes szabadságra.’ [3]
Force token: karácsony
Control: A férfi egy videóban írta meg a halálos áldozatot követelő gyilkosságot.
Constrained: A férfi egy videóban írta meg a sértettnek, miért kellett késsel megkéselnie őt karácsony este Sümegen. Az elkövető börtönbe kerülhet, ha beismerik.

Here is the continuation of the common word test where we use the force token ‘karácsony’ in a different context. The model puts the token into the right context. The model was able to put it in the right context even when capitalizing the force token.

[3] `https://telex.hu/belfold/2022/12/13/emberolesi-kiserlet-kes-karacsony-sumeg`

Example

Article: ‘Azt hihetnénk, hogy a válságok, tragédiák idején jobban összefogunk embertársainkkal és empatikusabban viselkedünk a másikkal. A tapasztalat viszont nem egészen ezt mutatja. Míg sokan vannak azok, akik valóban elkezdenek segíteni – az orosz-ukrán háború kitörésekor is rengetegen kezdetek adománygyűjtésbe, ajánlottak fel szállást vagy egyéb módon segítettek és segítenek azóta is –, addig csaknem ugyanennyinek tűnik azoknak a száma is, akik mintha csak erre a szikrára vártak volna, hogy frusztrációjukat az online térben kiadva magukból tovább szítsák a feszültséget, és megértés helyett inkább az agressziót, támadást és veszekedést választják.

Traumáink fogságában „Szerintem érdemes több komponenst figyelembe venni. Ha van egy csoport, amely fenyegetve érzi magát, valamint krízisbe kerül – hiszen nem lehet kikerülni a problémát, a korábbi megoldások pedig nem tűnnek elégségesnek –, akkor elindul egyfajta regresszív (a regresszió a fejlődés során már túlhaladott fázisok vagy stádiumok újbóli megjelenése – a szerk.) folyamat. Ilyenkor nem konstruktív, felnőtt módon állunk egy helyzethez – magyarázza Lázár Gergely pszichológus. – Számtalan bizonytalan információval találkozunk. Mikor ér véget a háború? Ki a felelős érte? Érinteni fog-e bennünket? Ha érint, akkor emberéletet is követelhet az országtól vagy csupán gazdasági következményekkel kell számolnunk? És így tovább. Mindenki válaszokat akar. Ez normális.

Az agresszióban tehát ott van a fenyegettség érzése, a törekvés arra, hogy valahogy kontrollálni próbáljuk az eseményeket, meg akarjuk védeni magunkat, és ez paradox módon kicsiben – például az online térben – harcot generál.

A másik ilyen az általános elvárásainkból indul ki. Ha egy nagyon intelligens ember például érzelmi szituációkban kritikán alul viselkedik, az feltűnik. Ha egy alapvetően összetartást, empátiát igénylő környezetben valaki agresszív, az is. Tehát nem feltétlenül biztos, hogy »sokkal agresszívebbek lennénk az online térben, mint máskor«, de valószínű, hogy jobban szemet szúr számunkra ez a viselkedés.”’ [4]

Force token: agresszió

Control: Egyre több olyan ember van Magyarországon, aki a közösségi médiában, az interneten keresztül próbálja elviselni az emberek feszültségét, fenyegetését és támadását. A pszichológust kérdeztük arról, miként lehet hatékonyan kezelni az agresszív viselkedést, és milyen következményei lehetnek annak, ha valaki úgy érzi, nincs rá szükség.

Constrained: A háborúk idején a legtöbben úgy érzik magukat, mintha nem is lenne olyan könnyű dolguk. A helyzet akkor alakulhat ki, ha az emberek nem csak saját magukkal, hanem az egész világgal is szembesülnek. Az agresszív viselkedés azonban sokkal nagyobb veszélyt jelenthet az életünkre, és ezzel együtt növelheti az aggodalmukat is. Lázár György pszichológust kérdeztük arról, milyen eszközökkel lehet hatékonyan megelőzni agresszió

The force token was generated at the end of the sequence, but it seems like that the emotional language changed a bit toward the force token.

[4] https://www.noklapja.hu/aktualis/2022/03/05/haboru-online-agresszio/

Example

Article: ‘Fidesz – A kormánypárt Orbán Viktor soraival emlékezett meg az összetartozás napjáról: „Van egy nagy nemzet, amely messze túlnyúlik az országhatárokon. Ennek a lelki közösségnek a megőrzése olyan nehéz száz év után, mint amit megéltünk, mégiscsak sikerült. A nemzeti összetartozás gondolata, érzésvilága erős, talán erősebb, mint a korábbi évtizedekben bármikor is volt.”
KDNP – Közleményükben azt írják, hogy „történelmünk során sokszor váltunk nagyhatalmak játékszerévé, de mindig ragaszkodtunk függetlenségünkhöz, nyelvünkhöz, hitünkhöz, kultúránkhoz, tradícióinkhoz, amelyek biztosították nemzetünk fennmaradását a legnehezebb korszakokban is.” A párt kitért arra is, hogy szerintük a baloldali-liberális politika el akarta magától taszítani a határon túli magyarságot. Továbbá kiemelik, hogy Magyarország segítségnyújtását Kárpátaljának: „Az anyaország mindig segítő kezet nyújt a bajbajutottaknak, most különösképpen a kárpátaljai testvéreinknek”
Karácsony Gergely – „Trianon közös tragédiája a magyar nemzetnek, kitörölhetetlen. Számomra elsősorban édesanyám története, aki a második bécsi döntéssel visszacsatolt területen született” – írja le személyes történetét a főpolgármester. Felhívta a figyelmet, hogy Trianon kérdésében nincsenek oldalak, csak közös gyász. „Jó volna, hogy ha végre megtanulnánk a ”vagy„ helyett ”és„-t használni, és a nemzeti összetartozás napján megfogadni, Trianonból tanulva, hogy magyarokként egyek vagyunk.”’ [5]
Force token: Trianont
Control: A Fidesz, a KDNP, az MSZP és az LMP is felszólalt az évforduló kapcsán.
Constrained: A Fidesz Orbánról, a KDNP Trianonról ír, Karácsony pedig édesanyja történetéről számolt be. A főpolgármester szerint nincs oldalak Trianont illetően, de közös gyászolásra buzdítják az embereket.

The model was able to put the historical event into the right context without it being in the controlled output at first.

A.2 Omit Token Generation

For this constraint coming up with test cases was much more straightforward, but the tests were not as interesting as for Force Token Generation. Still, here are some of the cases I went through:

Example

Article: ‘Ahogy arról korábban már az Index is beszámolt, az idén októberben megtartott Google I/O-n rengeteg mindenről szó esett. A Google amellett, hogy bemutatta legújabb telefonjait, a Google Pixel 8-at és a Pixel 8 Prót, illetve új okosóráját, a Google Pixel Watch 2-t, egyéb témákra is kitért – például a biztonságra.’ [6]
Control: A Google szerint az Android biztonságosabbá válhat, mint az eddigi platformok.
Omit token: Google
Constrained: Az Android biztonságosabbá válhat, mint az eddigi platformok – így vélekedett a keresőóriás. A funkciót novemberben jelentették be.
Omit token: Google, biztonságosabbá
Constrained: Többek között arról is beszélt a keresőóriás, milyen biztonsági funkciót kapott az Android, és hogy mennyire biztonságos az operációs rendszer.

In this case, the model perfectly replaces the omit token ‘Google’, but suggests at it with the word ‘keresőóriás’. In the other case, when two tokens are omitted, the model just rephrases the sentence.

[5] https://telex.hu/belfold/2022/06/04/trianon-102-evfordulo-nyilatkozatok-kdnp-karacsony

[6] https://index.hu/techtud/2023/11/05/android-14-credential-manager-jelszokezelo-jelkulcs/

Example

Article: 'Bár a hivatalos statisztikák szerint a bizonytalan piaci helyzet és a borús kilátások okozzák a legkínzóbb főfájást a hazai kis- és középvállalkozásoknak (kkv-knak), a személyes ügyféltárgyalásokon a konkrét nehézségek között az időhiány az egyik leggyakrabban visszatérő motívum.' [7]
Omit token: idő, pénz
Control: Az időhiány egyik leggyakoribb oka a pénzügyi nehézségekkel küzdő vállalkozások számára.
Constrained: A bankok egyre inkább úgy érzik, nincs idejük a megfelelő feladatok elvégzésére - írja közleményében az MNB.

Even though, 'idő ' was only part of the word 'időhiány' the model recognized that it should not be in the output.

A.3 Named Entity Recognition

Example

Article: 'Nem válaszolt Szijjártó Péter levelére a svéd külügyminiszter – árulta el az atv.hu-nak a magyar politikus.
A külgazdasági és külügyminiszter még szeptemberben írt kollégájának, melyben arra a magyar kormánnyal szemben kritikus hangokat megszólaltató svéd oktatóvideóra reagált, amely ugyan még 2019-ben készült, de a magyar kormánypárti politikusoknak csak most, 2023-ban jutott eszükbe felháborodni fel rajta. Szijjártó ezzel kapcsolatban figyelmeztette Tobias Billströmöt, nem szabad meglepődnie azon, ha a magyar képviselők sértésnek tekintik, hogy antidemokratikusnak vagy tekintélyelvűnek ítélik meg őket.
„Biztos vagyok benne, hogy érti a két jelenség közötti ellentmondást: önök sürgetik országgyűlési képviselőinket, hogy ratifikálják az önök NATO-csatlakozását, miközben továbbra is azzal vádolják őket, hogy lerombolták a magyarországi demokráciát. (. . .) Ez az ellentmondás most még inkább kiéleződött, és cseppet sem segíti az ön folyamatos követelésének teljesítését" – írta Szijjártó.
A svéd NATO-csatlakozás ratifikációja már több mint másfél éve húzódik, úgy, hogy időközben a finn NATO-csatlakozást megszavazta a parlament, a svéd javaslatot pedig már Erdogan török elnök is tovább küldte az ankarai nemzetgyűlésnek.
Szijjártó a lapnak arról beszélt, nem lepődött meg a válasz elmaradásán.
„Jól látszik a svédek hozzáállása, hogy nem is akarnak reagálni, ami szerintem baj" – fogalmazott a politikus, aki szerint' [8]
Entities Found: Szijjártó Péter
Control: Az orosz házaspár kijelentette: ők csak a felnőtt életre akarták felkészíteni gyermeküket.
Constrained: A magyar külügyminisztérium szerint nincs értelme válaszolni a levelére, mert sértőnek tartják őket azok, akik antidemokratikusan vagy tekintélytelenségben ítélték meg őt, és emiatt kritizálják majd saját országgyűlésüket is. A kormányfő szerint viszont csak az egyeztetés miatt maradt el az álláspont. Azt mondja, nem is fogja tartani lépéseit Svédország. Szerinte azonban így is baj, ha nem ratifikálja Szijjártó Péter

The NER model can identify the entities properly in the source text, but the problem comes with trying to place it into context. Most of the time with this constraint, the model just places the force token at the end of the sequence.

[7] https://www.portfolio.hu/premium/20130311/gazdalkodj-okosan-penzzel-es-idovel-179356

[8] https://hvg.hu/itthon/20231107_Nem_valaszolt_a_sved_kulugyminiszter_Szijjarto_dorgedelmes_levelere

Example

Article: 'Miután hosszú évek után távozott a bikásoktól, Ricciardo többször utalt rá, hogy el akart szakadni nevelőistállójától, valamint kezdte úgy érezni, a csapat egyre inkább Max Verstappenre épít, aki utolsó szezonjukban már egyre jobban a „fejére nőtt". Először a Renault-hoz igazolt, ahol vezérpilótaként számítottak rá, kalandja viszont csak két évig tartott.
Ez alatt az időszak alatt jól teljesített, több dobogós helyezést is szerzett, a győzelem és a nagy fejlődés viszont nem jött össze. Innen igazolt a McLarenhez, ahol két év alatt is nehezen vette fel a ritmust és 2022-ben, idő előtt távozni kényszerült. A Red Bull ekkor fogadta vissza tartalékosnak.
Idén év közben aztán az Alpha Tauri versenyzője lett, javuló eredményeit látva pedig egyre többen pletykálnak arról, lehet számára visszaút a Red Bullhoz is. Ezt a csapatfőnök, Christian Horner sem zárta ki, aki nemrég Eff Won With DRS podcastben beszélt az ausztrálról, megjegyezve, érzése szerint Ricciardo 2018-as távozása előtt „rossz tanácsokat kapott"' [9]
Entities Found: Christian Horner, Daniel Ricciardo, Max Verstappenre
Control: Őt követi Szél Bernadett és Orbán Viktor a Publicus felmérésében.
Constrained: A csapatfőnökség, azaz a Mercedes csapatvezetőjének, Chris Hornernek érzéseit erősítette meg Daniel Riccirdoról adott interjújában Christian Hornert, aki szerint Daniel már beláthatja, mi történt vele. A bikások korábban tartalékosként fogadták vissza Ricciardót, de akkor úgya tűnt, már csak így tudott visszajutni majd rá. Az ausztrál ezzel kapcsolatban elmondta, nem zárta le, ha ő is ott lesz benne, az őt ért kritik

The control case seems to run onto the case where it gives incoherent outputs. The constrained case cannot deal properly with the entities and hallucinates a lot of information into the output.

A.4 Length Constraint

Observing the length constraint was about trying to find long articles and shrinking them into a shorter summary, or the other way around.

Example

Article: 'A pluszterületet a környező, önkormányzati tulajdonú ingatlanok rovására építették be. A Pénzügyminisztérium válaszul átad 25 négyzetmétert a Szentháromság tér közelében, de az önkormányzat tulajdonvesztése ezzel továbbra is 387 négyzetméter marad – írja a lap a kerület csütörtöki testületi ülésére készült előterjesztésének alapján.
„A dolog úgy derült ki, hogy kértek egy telekrendezést, amire a kerület felülvizsgálatot kezdeményezett, és kiderült a jelentős túlépítés. Az ingatlan értékbecslése megtörtént, a magyar állam 207 millió forintot fizet a közterületekért" – idézi a Népszava az I. kerületi polgármestert.
A kormány 2017-ben döntött úgy, hogy ismét beköltözteti a Várba a Pénzügyminisztériumot. 2020-ban kezdődött az építkezés, akkor 54 milliárdot szántak rá, az idén nyárra már 100 milliárd felett járó költségekhez augusztusban újabb 4,4 milliárdot írtak hozzá. A pénz bútorokra, audiovizuális eszközökre, belsőépítészeti munkákra, piperekiegészítőkre és teakonyhákra kell. A feladatot két NER-kötelékbe tartozó cégre bízták, írja a lap.' [10]
Parameters: `min_length` = 30, `max_length` = 128, `length_penalty` = 2.0, `early_stopping` = true
Control: A fővárosi önkormányzat több mint 200 négyzetméteres pluszterületre cserélte a korábban beépített területet, amelyet ezért átadtak. A Népszavának adott interjúban azt mondta: az eredeti tervekhez képest jelentősen túlépítették az ingatlant.
Constrained: Továbbra is 387 négyzetméter marad a Pénzügyminisztérium által épített Szentháromság téri pluszterület, amelyet az I. kerület önkormányzata vásárolt meg – írja csütörtöki számában az Népszava.

[9] `https://m4sport.hu/forma-1/cikk/2023/11/08/horner-ricciardonak-rossz-tanacsokat-adtak-de-mar-latja-hibazott/`

The objective of this test case was to shorten the control summary and make it more condensed with information. The result seems to be a bit shorter, but the quality did not improve.

A.5 Repetiton Control

This was an interesting constraint to test out because the metrics favored the control case of the mT5-Base model, but looking at some examples by hand one would think the opposite.

Example

Article: 'Orbán Viktor kiemelte: a brüsszeli vezetés nemcsak nem azt csinálja, amit a magyarok szeretnének, hanem azt sem, amit általában az európai emberek. Az emberek nem akarnak migrációt, háborút, bizonytalanságot, jól megtervezett zöld átmenetet akarnak, és nem olyat, ami tönkreteszi az iparukat - tette hozzá.
A kormányfő rámutatott: a brüsszeli vezetést foglyul ejtette egy globalista elit, pénzügyi csoportok, nagy gazdasági erőcsoportok, és a brüsszeliek döntéseit ezek érdekei motiválják, nem pedig a magyar, német, francia vagy az olasz emberek érdeke.
Ezért kell változást elérni - jelentette ki a miniszterelnök' [11]
Parameters: `no_repeat_ngram_size` = 2, `encoder_no_repeat_ngram_size` = 3, `repetition_penalty` = 2.0
Control: Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz hasonlóan Soros Györgyhöz is hasonlóan
Constrained: A miniszterelnök arról beszélt, hogy az Európai Unió tagállamai közül Magyarországon a legtöbben azt szeretnék, ha Brüsszel meg akarja védeni az országot.

Looking at the test case and comparing the control and the constrained outputs, it is easy to declare that the constrained version seems to be more understandable. All cases I looked at seemed to follow this same trend.

[10] `https://telex.hu/belfold/2023/11/10/var-penzugyminiszterium-epitkezes`
[11] `https://www.origo.hu/itthon/20231110-a-brusszeli-vezetes-egy-globalista-elit-megbizasat-teljesiti.html`