# The impact of language family on D2T generation in under-resourced languages.

MSc Thesis

Georgios Christopoulos
Student number: 2789175
g.christopoulos@students.uu.nl

July 9, 2024

1st supervisor: Dr.Albert Gatt
2nd supervisor: Dr.Antal van den Bosch

Department of Computer Science
Faculty of Science

Universiteit Utrecht

# Contents

## Acknowledgements

First and foremost, I would like to express my gratitude to my supervisors, Dr. Albert Gatt and Dr. Antal van den Bosch, for their continuous support, guidance, and encouragement throughout my research and the writing of this thesis. Their expertise and insights have been invaluable, and this work would not have been possible without them.

I am also immensely grateful to my partner Ioana for her continuous support, help, and belief in me. I would also like to thank my best friend, Konstantinos, for his encouragement, advice, and feedback during the MSc thesis. Lastly, I thank my family for supporting me and making the MSc thesis possible.

# Abstract

The paper delves deep into the challenges and methods of generating text from structured data(RDF triples) for under-resourced languages based on the WebNLG challenge. The main question of this paper is to assess how important language families are in helping the model generate text without prior examples(zero-shot) in the WebNLG target languages. In the paper, we work with limited resources. We utilize an already pre-trained encoder-decoder LLM, such as mT5-small, to test the hypothesis, and we train as much as necessary before we notice a plateau in performance since there are hardware limitations. By applying further pre-training and testing different finetuning strategies, the aim is to improve text coherence and fluency and assess how well the model extracts the information from the RDF triples. As part of our ablation experimentation, we pre-train and finetune to assess their impact on the D2T task. The experimentation starts with the simplest model, pre-trained on the OPUS-100 dataset and finetuned on the English WebNLG dataset. The pre-training recipe remains the same, but for the finetuning step, the WebNLG dataset is altered to include more linguistically diverse language samples. Lastly, we introduce an augmentation technique to further alter the WebNLG dataset and generate samples for all the relative languages we are trying to target. In the end, the best finetuning strategy is applied to a clean mT5 model to assess the influence of the pre-training. Later on, in the meta-experiments, we generate augmented data for the languages we target in WebNLG, which takes the models out of the zero-shot setting. With these extensive experiments, we try to measure model performance using automatic metrics, involving manual analysis, and lastly, a comparison with other models in the WebNLG 2023 challenge. For the assessment, we use automatic metrics such as BLEU, ROUGE, METEOR, TER, chrF++, BertScore, and PARENT to provide a more holistic view of our model's capabilities. In a few words, this study aims to contribute in the following ways:(a) What is the influence of language families under a zero-shot setting? (b) is further pre-training necessary, or does it tend to have diminishing results? (c) Does finetuning with noisy data provide any benefit? Lastly, (d) How does our model compare with the other models of the WebNLG 2023 challenge?

# 1 Introduction

In recent years, the emergence of capable Artificial Intelligence (AI) models has led to the development of general task solvers. These models can solve many tasks with little or no examples, a feat previously thought unimaginable. The popularity of these models increased with the emergence of Transformer models and their applications, such as Chat-GPT [1], which is a chatbot trained on large corpora from various sources, enabling them to acquire general knowledge and perform tasks such as question answering, coding, etc. The development of more advanced generative models was also on its way to train even larger and more capable task solvers. These models are known as Large Language Models (LLMs) and have emerged due to years of research in Natural Language Processing (NLP).

NLP is focused on enabling computers to understand, interpret, process, and generate human language meaningfully and usefully. Similarly, Natural Language Generation (NLG), a subfield of NLP, focuses on generating natural language from structured or unstructured data. The style and quality of the text are also important aspects of NLG, intending to generate personalized and readable text that is coherent in semantically diverse content scenarios.

Generating text from structured data is challenging, especially when dealing with under-resourced languages. Recent research (1) indicates a significant gap among the over 7,000 world languages in terms of available resources in the field of NLP. Although there have been some advances in inclusion for under-represented languages at major NLP conferences, linguistic diversity and inclusion still pose a challenge. Thus limiting the language representation online and its inclusion in datasets.

Even though there are no language-universal models, a viable approach to this problem is multilingual LLMs; these models are trained in multiple languages and can produce text in a wide range of languages they have been exposed to during training. Efforts like the WebNLG challenge (2) combine data-to-text generation with under-resourced languages. The WebNLG challenge provides a benchmark dataset in which participants convert non-linguistic data from the Semantic Web into textual output. Initially focused on English, the challenge now includes Irish, Maltese, Breton, and Welsh.

This paper aims to provide valuable insights into how existing data can improve the generation of under-resourced languages in the context of the 2023 WebNLG challenge on under-resourced languages. Our approach involves using a popular multilingual model such as mT5 and further enrich-

[1]https://openai.com/chatgpt/

ing its general knowledge about the family languages of the target languages (WebNLG languages) using the OPUS-100 dataset. The aim is to enable our text-to-text model to process structured data (RDF triples) and directly generate text in under-resourced languages. Also, we are planning various experiments with models undergoing further pre-training and many finetuning strategies to measure performance and generation quality. With a few words, our research aims to contribute to and answer the following research questions:

- Main research question:

  Q1: How can language families affect the NLG process?

- Following subquestions:

  SQ1: Is further pre-training necessary, or does it have diminishing returns?

  SQ2: How does finetuning with noisy data impact the performance of mT5 in the text generation tasks?

  SQ3: How does our solution compare to other participating WebNLG models?

The rest of the paper is structured as follows: Section 2 provides an overview of the related work in the field. Section 3 explains our methodology in detail, including the implementation of experiments and our choices. Section 4 presents the results of our experiments and compares them with other WebNLG contested models. Section 5 briefly discusses the problems encountered and their contribution to the NLG research field. Lastly, Section 6 discusses the future direction of the project. All the code that was used for this project can be found on GitHub [2]

# 2 Related work

This section will discuss data-to-text generation in detail, including the most used datasets, the main subtasks, end-to-end architectures, and modular pipelines. We also compare the modular and end-to-end systems. Then, we discuss what large language models(LLMs) are and their architectures, and we discuss scaling laws and capabilities that are unlocked as the model size grows. We touch upon the pre-training of LLMs as well as the

[2] https://github.com/GeorgiosChristopoulos96/Thesis

tuning strategies. Moreover, we mention the current state-of-the-art multilingual models. Lastly, we touch upon the top-performing models from the WebNLG challenge, which contributed to the modeling choices we decided to implement in this paper.

## 2.1 Data-to-text generation

Data-to-text generation broadly refers to automatically producing text from non-linguistic input (3; 4).

Data-to-text (D2T) generation has historically employed modular pipeline architectures, converting non-linguistic input data into natural language through several intermediate steps (5; 4). However, recent advancements have seen the emergence of neural models proposing end-to-end approaches, directly rendering non-linguistic input into natural language with minimal intermediate representations. In the following subsections, we introduce some datasets used in the D2T generation task.

### 2.1.1 NLG main subtasks

Reiter and Dale (6; 3) decompose the process of generating text in 6 subtasks. In NLG systems usually, the following six are found:

1. **Content determination:** Deciding what information should be communicated in the text.
2. **Discourse planning:** Imposing ordering and structure over the messages to be conveyed.
3. **Sentence aggregation:** Determining how to group the information into sentences.
4. **Lexicalization:** Deciding the words and phrases that should be used to express information.
5. **Referring expression generation:** Selecting words or phrases to identify domain entities
6. **Linguistic realization:** Producing text that follows language rules to be syntactically, morphologically, and orthographically correct.

**Content determination**: It is the first step of the NLG system, and it involves deciding what information to include in the text. Although content determination is present in almost every *NLG* system (7), generalizing content determination can be challenging because it depends on the target application's details. For example, this process can be influenced by the system's data, domain knowledge, and user models. For example, the determination of the content of a rail system differs from that of customer support. However, all these systems share the need to process the input data (filtering, summarization).

Some systems use deep reasoning, analyzing user goals to identify necessary information. Others rely on experts' domain-specific rules, which are easier to implement and adapt to legal or bureaucratic needs. Techniques for establishing content rules include analyzing text corpora and consulting with domain experts to develop relevant message classes and conditions for their inclusion. The rules for content determination can be seen as a form of knowledge acquisition (KA) (8) since many steps in the process of KA can be applied to content determination.

**Discourse planning**: This process involves organizing content into a coherent text. After determining what information to communicate, discourse planning decides how to structure these messages effectively. It uses a text plan structured like a tree, where messages are the leaves and internal nodes show how messages are grouped conceptually. The plan outlines discourse relations between messages, indicating how text fragments relate to each other, such as through elaboration or contrast, often highlighted by specific cue words. There are two approaches to discourse planning. The first one is the planning-based approach. It aims to organize content messages into coherent wholes, using AI-style planning operators to define the conditions and effects of applying discourse relations. Our understanding of discourse relations still limits this sophisticated method and is computationally expensive, making it less common in real-world applications. The other approach is schema-based, which offers a more practical alternative for specific domains, identifying text patterns through analysis and expert consultation. These approaches use schemas—defined patterns for constructing text plans—to structure messages and relations (9). Schemas can request specific content 'on demand,' allowing for an interplay between content determination and discourse planning. This method, often implemented via specialized programming constructs, promises easier development of domain-specific NLG systems, though many developers create their own schema languages due to the lack of standardization.

**Sentence aggregation**: Sentence aggregation aims to combine multiple messages into a single, coherent sentence or text plan (10). Sentence aggregation involves deciding which messages to combine and determining the syntactic mechanism for their combination. It also aims to enhance text readability and fluency without altering the informational content. There are many sentence aggregation strategies, such as no aggregation, in which each message is realized as a separate sentence, possibly with pronominalization to improve coherence. Another one is to combine messages using a relative clause. This formation involves attaching a relative clause to a sentence to incorporate additional information without starting a new sentence. The third one is simple conjunction, which uses conjunctions like "and" to connect sentences, conveying more than one message in a single sentence. There are also aggregation formations, such as the ellipsis one, which eliminates repeated constituents when sentences have common elements, making the text more concise. The set formation aggregates messages with a common action but different objects into a single sentence that lists all objects. Lastly, embedding formation is about embedding one clause as a constituent of another, often through relative clauses (6).

But to create effective aggregation rules, these rules can be derived from psycholinguistic research, writing guides, or by analyzing patterns within specific corpora to reflect genre-specific conventions. A general constraint involves aggregating only sibling nodes in a text plan to maintain coherence, although a weaker version may allow for the aggregation of all descendants of an internal node under certain conditions

**Lexicalization**: Lexicalization is selecting specific words or phrases to accurately and effectively express domain concepts or relations. This task is pivotal for crafting text that communicates intended meanings in a clear, engaging, and context-appropriate manner. Advanced graph-rewriting algorithms are crucial in lexicalization, transforming input graphs of domain concepts into linguistic expressions through mappings defined in comprehensive dictionaries. This approach is especially valuable in multilingual NLG, allowing for the expression of the same conceptual content across different languages and accommodating lexical divergences naturally. Decision trees and rule-based methods guide lexical choice in practical applications, enhancing text variety and adapting language use to match contextual or stylistic requirements. Through careful lexicalization, NLG systems achieve fluency and readability, producing texts that convey information accurately and reasonably to the intended audience, demonstrating the interplay between domain knowledge and linguistic expression (3; 6).

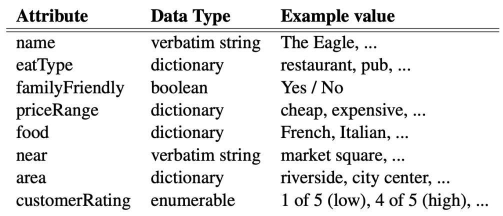
**Referring Expression Generation**: Referring expression generation involves crafting descriptions to identify target entities within a discourse context unambiguously (11). It varies in complexity based on the information needed to distinguish the entity from others, influenced by the discourse context. Initial references to objects might simply use their names or describe their physical locations if they are uniquely identifiable that way. Subsequent references often employ pronouns when the entity was mentioned recently, and there's no ambiguity with other entities or definite descriptions that may incorporate base nouns and additional modifiers to differentiate the target entity from others mentioned in the discourse.

Research has explored various strategies for generating these references, with initial introductions receiving less focus than pronouns and definite descriptions. Using pronouns is crucial for avoiding repetition and maintaining coherence, relying on algorithms that ensure clarity and prevent misinterpretation by considering the entity's recent mention and the absence of potential referential ambiguity. For definite descriptions, methods include starting with a basic noun and adding distinguishing features as needed, a practice supported by corpus analysis within specific application domains. This approach helps in situations where multiple entities of a similar type are discussed, ensuring clear and precise identification of the intended target.

**Linguistic realization**: Linguistic realization involves encoding grammatical knowledge to generate sentences that communicate messages accurately, considering syntax, morphology, and the peculiarities of a language. It can involve various approaches, such as the inverse of parsing, systemic grammars, meaning-text grammars, or using templates, each with its own method for transforming semantic content into grammatically correct text. Realization is typically seen as converting abstract representations of messages into coherent and grammatically correct sentences. One common approach is using templates, which can be straightforward when the variability of the text is minimal. Templates predefine sentence structures where variable slots are filled with relevant data. This method ensures grammaticality and is effective in controlled domains but lacks flexibility and adaptability to more complex linguistic needs (6; 12). Another approach involves hand-coded grammar-based systems, which use rules from linguistic theories to generate text. These systems handle various linguistic phenomena and produce more nuanced outputs than template-based systems, utilizing frameworks like systemic-functional grammar and lexicalized tree adjoining grammar (13; 14). Statistical approaches have become prominent with machine learning techniques. These methods train models on large corpora to predict word sequences from input representations, capturing linguistic variability and handling complex text generation tasks (15; 16). Modern approaches integrate these methods, using templates for fixed structures and statistical models for flexibility. This hybrid approach balances templates' reliability with statistical methods' adaptability (7).

### 2.1.2 D2T datasets

**E2E**: The E2E dataset (17), designed for training end-to-end, data-to-text applications in the restaurant domain, includes over 50,000 English verbalizations corresponding to about 5,751 dialog-act-based meaning representations (MRs). These MRs contain 3-8 attribute-value pairs from 8 attributes, which an example can be found in figure 1. The dataset was created using CrowdFlower, using pictures as stimuli for data collection, which yielded more natural and informative human reference phrases. The dataset is divided into training, validation, and testing sets in a 76.5-8.5-15 ratio, ensuring distinct MRs across sets and maintaining a balance in MR and text length distribution. While contributors were encouraged to include all information from the MRs, omissions were not penalized, making the dataset suitable for studying content selection in pipeline data-to-text systems. This setup results in a rich corpus with higher lexical and syntactical variation than similar corpora.



| Attribute | Data Type | Example value |
|---|---|---|
| name | verbatim string | The Eagle, ... |
| eatType | dictionary | restaurant, pub, ... |
| familyFriendly | boolean | Yes / No |
| priceRange | dictionary | cheap, expensive, ... |
| food | dictionary | French, Italian, ... |
| near | verbatim string | market square, ... |
| area | dictionary | riverside, city center, ... |
| customerRating | enumerable | 1 of 5 (low), 4 of 5 (high), ... |

***Figure 1.*** *Ontology of a specific domain of the E2E dataset*

**DART**: The DART dataset (18) is designed for text generation from structured data records, specifically using RDF triplets. It comprises 82,191 examples across various domains, where each input is a semantic RDF triple set derived from data records in tables and the schema's tree ontology, annotated with sentence descriptions. Unique for its hierarchical, structured format and open-domain nature, DART stands apart from other table-to-text corpora. The primary task associated with DART is RDF-to-text generation, evaluated on metrics like BLEU (19), METEOR (20), BLEURT (21), TER (22), MoverScore (23), and BERTScore (24), with a benchmark model (BART-Large) (25) achieving notable scores in these areas. The dataset, entirely in English, includes data instances with annotations (text descriptions and sources), a subtree_was_extended boolean indicator, and RDF triplets. It is split into training, validation, and test sets with 30,526, 2,768, and 6,959 examples. DART is curated to enhance the accessibility of knowledge bases for lay users, drawing from varied sources such as WikiTableQuestions (26), WikiSQL (27), WebNLG (28), and Cleaned E2E (29), and involves a comprehensive two-stage annotation process combining skilled annotator input and broader group annotations for sentential descriptions.

**WikiBio**: The WikiBio dataset (30) is a comprehensive collection designed for text generation tasks, specifically to convert structured data into textual descriptions. It focuses on producing the initial sentences of Wikipedia biographical articles based on infobox data. The dataset, comprising over 700,000 articles, was prepared by tokenizing sentences, converting numbers to tokens (except years), and lowercasing text for uniformity. It supports machine learning research, notably in neural language models and automatic text generation, by offering a rich source of structured information paired with corresponding narrative text.

### 2.1.3 End-to-end architectures

End-to-end approaches in data-to-text (D2T) generation have significantly evolved with deep learning and neural network architectures. Unlike traditional modular approaches that rely on separate components for content selection, structuring, lexicalization, and realization, end-to-end methods aim to generate textual output directly from data inputs using a single neural model. These approaches have been favored for their ability to learn complex mappings from data to text without requiring intermediate representations or hand-crafted rules. Moreover, in section 2.2, we discuss large language models favored recently for text generation tasks.

**Long Short-Term Memory** Early end-to-end systems often utilized RNNs, especially Long Short-Term Memory (LSTM) networks (31), which was a novel architecture designed to overcome the limitations of traditional recurrent networks(RNN) in learning long-term dependencies, due to their capability to handle sequential data and remember long-term dependencies. LSTM incorporates special units called memory cells, along with input and output gates, to regulate the flow of information, allowing it to mitigate fading gradient issues. These models were trained to generate text one token at a time, conditioned on the input data and the previously generated tokens, making them suitable for tasks like generating weather reports or financial summaries from structured data.

**Sequence-to-Sequence (Seq2Seq) Models** The Seq2Seq architecture, which typically combines an encoder RNN with a decoder RNN, became a cornerstone for machine translation tasks, which later was adapted for D2T generation, specifically with LSTM networks (32). The method involves mapping input sequences to fixed-dimensional vectors with one LSTM and decoding the target sequence from these vectors with another LSTM, showcasing the ability to handle sequence-to-sequence learning tasks with minimal assumptions about sequence structure. This approach achieved notable results on a large-scale English-to-French translation task, outperforming standard phrase-based Statistical Machine Translation (SMT) systems. Later, as seen in the work of Ferreira et al. (33), they adapted the Seq2Seq model for the Abstract meaning representations-to-text task, framing it as a translation task. Later, Bahdanau et al. (34) introduced the attention mechanism, which would set up the foundation for the Transformer models that were introduced in the work of Vaswani et al. (35). In section 2.2.3, we discuss the Transformer models and the attention mechanism in greater detail.

**Pre-Trained Language Models** Radford et al. (36) introduced a novel approach to natural language understanding by pre-training a language model followed by discriminative finetuning on specific tasks. By pre-training on a diverse corpus of unlabeled text, the model learned a universal representation that improved performance across various language understanding benchmarks, outperforming task-specific discriminatively trained models in most cases. They used the Transformer architecture due to its effectiveness in handling long-term dependencies, with minimal changes needed for task-specific adaptations. The results showed substantial improvements in question answering, textual entailment, and document classification, establishing new state-of-the-art performance.

Devlin et al. (37) introduced BERT (Bidirectional Encoder Representations from Transformers), a new method for pre-training language representations (see section2.2.1) that outperforms existing models across a wide range of NLP tasks by pre-training deep bidirectional representations from unlabeled text. BERT's architecture was unique because it used a multi-layer bidirectional Transformer encoder conditioning both left and right context in all layers, allowing it to be finetuned with just one additional output layer to create state-of-the-art models for a diverse array of tasks, including question answering and language inference without significant task-specific modifications.

Brown et al. (38) introduced GPT-3, an autoregressive language model with 175 billion parameters, and explored its performance in a few-shot learning (see section 2.2.2) setting without task-specific finetuning. Scaling up language models significantly enhances their ability to perform a wide range of NLP tasks, including translation, question-answering, and cloze tasks, by merely providing a few examples or instructions in natural language. GPT-3 demonstrates competitive results across various benchmarks, sometimes outperforming state-of-the-art models that underwent task-specific finetuning. However, the large pre-trained models struggle with potential biases from training on large web corpora.

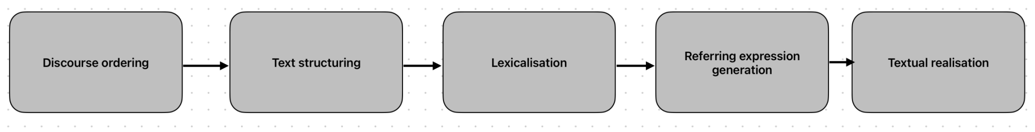


***Figure 2.*** *Ferreiras' modular pipeline inspired by Reiter and Dale (4)*

### 2.1.4 Modular pipelines

Ferreira et al. (39) systematically compared neural pipelines in modular subtasks and end-to-end approaches for generating text from RDF triples. The comparison validated whether generating text through a modular pipeline with discrete steps can outperform end-to-end systems, which do not have intermediate representations but mostly map the RDF triples to the corresponding output text.

In figure 2, we can see the authors' implementation inspired by Reiter and Dale (4). The first subtask is discourse ordering. Discourse ordering determines how the communicative goals(RDF triples) should be verbalized. The second subtask is text structuring, which organizes the ordered triples into paragraphs and sentences. The third subtask, lexicalization, aims to find the proper phrases and words to express the content that must be included in each sentence. The fourth subtask, referring to expression generation, generates references to the entities of the discourse. This template outlines where and how to refer to specific entities, ensuring that the references are correctly placed and entities are accurately identified within the discourse. The last task is textual realization, which finalizes data conversion into text by adjusting verbs and determiners to their correct forms, ensuring grammatical accuracy.

Puduppully, Dong, and Lapata (40) implemented a neural network architecture incorporating content selection and planning without sacrificing end-to-end training. Their pipeline incorporated content selection and planning to maintain coherence over long texts, ensure a logical ordering of facts, and avoid redundancy and inaccuracies. The first stage of content selection is seen in figure 3, which helps the model to understand how records relate by using a content selection gate mechanism. It calculates attention scores across records to determine their relevance. An attentional vector is formed for each record, combining its information with related records. The content selection gate then adjusts the record's representation based on contextual importance, allowing the model to focus on the most relevant details for generating summaries. The second subtask is content planning, which assists the model in generating structured game summaries by learning explicit content plans from the training data, which guide what information to include and in what order. Since datasets like RotoWire (41) don't have inherent content plans, these are created by mapping summary text to input data elements. The model uses Pointer Networks (42) with an LSTM decoder to generate a sequence pointing to input records, determining the structure of the output text. This approach helps generate coherent, well-structured summaries by specifying the sequence of discussed entities based on learned content plans.

Lastly, the last subtask is about text generation. In this step, The model predicts the output text by considering both the input data and the planned content structure. It uses a bidirectional LSTM to encode the content plan and an LSTM-based decoder for text generation, with attention to focus on relevant parts of the plan. Additionally, it employs a copy mechanism that directly includes input data terms in the output, enhancing accuracy and relevance. The planned content and the ability to copy directly from inputs enable the generation of detailed and contextually accurate text.

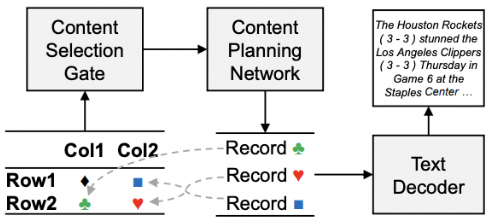

Content Selection Gate
Content Planning Network
The Houston Rockets ( 3 - 3 ) stunned the Los Angeles Clippers ( 3 - 3 ) Thursday in Game 6 at the Staples Center ...
Col1 Col2
Row1
Row2
Record ♣
Record ♥
Record ■
Text Decoder

***Figure 3.*** *Puduppullys' pipeline*

### 2.1.5 Neural and Modular architectures comparison

As mentioned previously, Ferreira et al. (39) tried to compare and address issues of the end-to-end systems by comparing them with a modular system with well-separated individual steps. The traditional pipeline used by Ferreira, inspired by Reiter and Dale (4) steps, usually entails discourse ordering, text structuring, lexicalization, referring expression generation, and textual realization. Evaluating these systems suggests that having explicit intermediate steps, such as the modular pipeline in the generation process, results in better texts than the ones generated by end-to-end approaches since they better describe the data on all domains of the corpus. Also, the pipeline models generalize better to unseen domains, whereas the performance of the end-to-end system drops significantly. The qualitative analysis showed that end-to-end generated texts have the problem of Hallucinations (43), which means they describe non-linguistic representations that are not present in the input

Puduppully, Dong, and Lapata (40) concentrated on content selection and planning to improve fluency in natural language generation since current end-to-end approaches are not explicitly modeled on what to say and how to say it. Their approach aimed to generate more organized, interpretable, and faithful text by explicitly modeling what information to include and in what sequence. In their evaluation, they focused on how well the model selected relevant content, generated accurate relations, and ordered content correctly. Results showed that the Neural Content Planning (NCP) model outperformed encoder-decoder models in all metrics, demonstrating improvements in content selection, relation generation, and ordering.

## 2.2 Large Language Models

Large language models (LLMs) usually refer to the Transformer language models that contain hundreds of millions or even billions of parameters, which are trained on massive text data (44), such as GPT-3 (38), PaLM (45), BLOOM (46), OPT (47), LLaMA (48), Chinchilla (49), BART (25) and T5 (50). LLMs exhibit strong capacities to understand natural language and solve complex tasks via text generation.

Of course, these models only emerged after a while. There has been a progression starting from early statistical models such as the n-gram model (51), which was based on the Markov assumption, meaning predicting words based on previously fixed context, to task agnostic feature learners such as Word2Vec (52), then progressing to the transferable NLP task solvers such as ELMo (53), BERT (37), GPT-1 (36) and GPT-2 (54). Lastly, the progression reaches today's models, advanced transformer-based models, such as GPT-3 (38) and GPT-4 (55), which are general-purpose task solvers. In the following subsections, we categorize the models based on their architectures and give an overview of how scaling can unlock new capabilities in the model. Then, we look at how pre-training equips the model with general knowledge and tuning strategies to elicit certain abilities in the LLMs.

### 2.2.1 Encoder models

An encoder is a transformer block with multiple layers stacked together and can include an attention layer, normalization layer, feedforward layer, and residual connections, which are tailored for capturing and processing the input data into a representation that captures the information (35). The encoder-only models are not suitable for text generation but primarily for

NLP tasks such as Sentiment Analysis, Named Entity Recognition (NER), Question Answering, Language Inference, and Text Classification (37). BERT is the most popular encoder model (Bidirectional Encoder Representations from Transformers) (37). BERT is designed to pre-train deep bidirectional representations by conditioning on both the left and right context in all layers. The objective of BERT is masked language modeling (MLM) and next-sentence prediction (NSP). During MLM, random words in a sentence are replaced with a [MASK] token, and the model is trained to predict the original word based on the context provided by the other words in the sentence. This process enables BERT to understand the context in which a word appears and generate embeddings that capture this contextual information. Because the BERT model is context-sensitive, the produced embeddings were a leap in the quality of word representations. Moreover, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for various tasks, such as question answering and language inference, without substantial task-specific architecture modifications. Other encoder-based models have emerged due to BERT.

One of them is DistilBERT (56), which is a smaller, faster, cheaper, and lighter version of BERT. DistilBERT is designed to retain 97% of BERT's performance while being 40% smaller and 60% faster. It is an encoder-only model that can be used for a wide range of NLP tasks, providing an efficient alternative to BERT for environments with limited computing power.

Another is RoBERTa (Robustly optimized BERT approach) (57). This model builds on BERT's architecture and modifies key hyperparameters, removing the next-sentence pre-training objective and training with much larger mini-batches and learning rates. It demonstrates improved performance over BERT on several benchmark NLP tasks.

Lastly, ALBERT (A Lite BERT) (58) is a model that offers a light version of BERT that introduces two parameter-reduction techniques to lower memory consumption and increase the training speed of BERT. It achieves comparable or superior performance on benchmark NLP tasks with significantly fewer parameters than BERT.

### 2.2.2 Decoder models

A Decoder (38) is a transformer block similar to the encoder. It does not allow attention to go further than the current token using masking of those tokens to maintain its autoregressive nature. Decoders typically use multi-head or cross-attention attention to capture the information and process the encoded representations of input data into the desired output format. The model generates new tokens by predicting each token in a sequence based on the tokens that precede it, step by step, crafting coherent and contextually sound outputs. This text-generation process, which depends on the previous content, is called autoregressive generation. The decoder-only models have been used in many tasks, such as text generation, summarization, code generation, question answering, and dialogue systems. Below, we discuss some prevalent models that use a decoder-only architecture.

OpenAI released the GPT-3 in June 2020, which brought advanced NLG capabilities, and it was a massive scale-up from the GPT-2 model, setting a new standard for language model sophistication (51; 38). Brown et al. (38) introduce the concept of language models as few-shot learners, where a model trained on a wide variety of data uses this broad knowledge to adapt to new tasks at inference time without finetuning. It contrasts three approaches: FineTuning (FT), where a model is updated with a task-specific dataset; Few-Shot (FS), where the model is given a few task demonstrations at inference time; and One-Shot (1S), which is like few-shot but with only one demonstration. The focus is on evaluating GPT-3's task-agnostic performance. It explores the concept of "few-shot learning" in language models, discussing their capabilities and limitations in learning from a limited number of examples.

Later, Google Deepmind released Chinchilla (49), a model focused on optimizing the model size, data, and computing trade-offs. The Chinchilla paper investigates the relationship between language model size, dataset size, and training tokens. It reveals that large models like Gopher are often undertrained and that model size and data volume should be scaled proportionately for optimal results. The study finds that using equal computing resources with fewer parameters and more data helps Chinchilla to significantly outperform larger models like Gopher (59), GPT-3 (38), Jurassic-1 (60), and Megatron-Turing NLG (61). It also discusses the importance of setting learning rate schedules to match training tokens for the best performance. The improvement that Chinchilla demonstrated is that, with optimal training data size and computing, smaller models could achieve performance comparable to larger models, emphasizing efficiency in model training.

Google Research in 2022 introduced the Pathways Language Model (PaLM), a large-scale Transformer language model with 540 billion parameters (45). PaLM is notable for achieving state-of-the-art results in few-shot learning across a wide array of language understanding and generation benchmarks. It demonstrated significant improvements in various English NLP tasks over prior models such as GPT-3, Megatron–Turing NLG, Gopher, Chinchilla, and LaMDA (62). Moreover, PaLM showed remarkable performance on multilingual NLP benchmarks, even though only a fraction of its training corpus was non-English.

The OPT model is another prevalent example with a focus and commitment to open-source accessibility, which was released in 2022 by Meta AI. The OPT paper by Susan Zhang et al. (47) discusses evaluations of Transformer language models of various sizes, from 125 million to 175 billion parameters, and their performance on standard NLP tasks. The study follows the experimental setup of GPT-3 and compares results across multiple tasks, observing that while performance broadly aligns with GPT-3, it can vary significantly between tasks. It also highlights the unpredictability of model performance on smaller validation sets and discusses inconsistencies when replicating GPT -3's results, indicating possible methodological differences.

On the same line of thought, BigScience, an open research collaboration, released BLOOM, a 176-billion-parameter multilingual language model. The initiative focused on releasing a public LLM focused on ethical AI, open science, and language inclusivity. BLOOM is an open-source model with comprehensive documentation and a license promoting responsible use. The model is evaluated in zero-shot and few-shot settings, showing superior performance in multilingual summarization and code generation compared to other LLMs. The performance of BLOOM is attributed to its training on multilingual-focused data. The paper also addresses the challenges and ethical considerations in developing large-scale models, including data selection and model alignment with human values. Achieved higher performance than OPT (47), and the model gets better the larger it gets.

Lastly, Meta AI released on February 2023, the LLaMA(Large Language Model Meta AI) model (48). The LLaMA model focused on its efficiency and optimizations within computing budgets, introducing language models that prioritize computational efficiency. It used smaller models trained on extensive publicly available data in languages with Latin and Cyrillic scripts. It is built upon the transformer architecture with improvements for stability (RMSNorm), non-linearity (SwiGLU), and positional embeddings (RoPE). The training utilizes the AdamW optimizer with specific hyperparameters and a cosine learning rate schedule, emphasizing the relationship between model size, data volume, and computational resources (48; 51).

### 2.2.3 Encoder-decoder models

As we show in section 2.1.3, Sutskever et al. (32) introduce a framework for machine translation by using Seq2Seq(RNN encoder-RNN decoder) that will influence later work on developments such as the attention mechanism and the Transformer models, which are discussed below. Bahdanau et al. (34) introduced a novel approach to neural machine translation that significantly improves the traditional encoder-decoder architecture by integrating an alignment mechanism, allowing the model to focus on relevant parts of the input sentence dynamically during translation. This mechanism is called attention. The attention mechanism addresses the limitations of fixed-length context vectors, particularly in translating long sentences, by letting the model search for and attend to specific parts of the source sentence for each target word. In their research, the proposed RNNs demonstrated superior performance to the standard RNN encoder-decoder models and traditional phrase-based statistical machine translation systems, especially in handling longer sentences.

Later, Vaswani et al. (35) introduced the Transformer models, a novel neural network architecture for Seq2Seq models that relies solely on attention mechanisms, excluding traditional recurrent or convolutional layers. This implementation significantly increased training parallelization, leading to a faster and more efficient learning process. The Transformer achieved state-of-the-art results on the WMT 2014 English-to-German and English-to-French translation tasks, outperforming existing models and ensembles with considerably lower training costs. Additionally, the model demonstrated its versatility and generalizability by performing well in English constituency parsing tasks, further establishing its effectiveness beyond translation.

With the current state-of-the-art LLM models of ever-growing size, FacebookAI released BART (25) in 2019, which stands for Bidirectional and Auto-Regressive Transformers. BART is a transformer-based model, and it is trained by corrupting text with an arbitrary noising function and then learning a model to reconstruct the original text for natural language processing (NLP) tasks, such as comprehension, translation, summarization, and Natural Language Generation(NLG). BART uses a standard Transformer-based NMT architecture and can be seen as generalizing BERT (with its bidirectional encoder) and GPT (with its left-to-right decoder). It's effective for text generation and comprehension tasks, achieving state-of-the-art results in various applications. It matches the performance of RoBERTa (57) with comparable training resources on GLUE (63) and SQuAD (64) and achieves new state-of-the-art results on a range of abstractive dialogue, question answering, and summarization tasks. BART performance also increases over a back-translation system for machine translation, with only the target language pre-training.

At the same time, the Text-to-text transfer transformer (T5) (50) is a model developed by Google AI Language in 2019 with a primary focus on transfer learning in the domain of natural language processing (NLP). The model is trained to denoise corrupted spans of text. In transfer learning, a model is first pre-trained on a data-rich task in which it acquires general-purpose knowledge before being finetuned on a downstream task. This approach has emerged as a powerful technique in natural language processing (NLP), widely adopted by present-day state-of-the-art LLMs. T5 is a unified framework that converts all text-based language problems into a text-to-text format, enabling consistent pre-training objectives, architectures, and transfer approaches across a wide range of language tasks such as machine translation in which the model did not achieve state-of-the-art results by being pre-trained only in English. The study includes extensive experiments with various model architectures, unsupervised objectives, pre-training datasets, and transfer strategies. It introduces the Colossal Clean Crawled Corpus (C4) as a pre-training dataset and experiments with models up to 11 billion parameters. T5 achieves state-of-the-art results on numerous benchmarks and releases datasets with pre-trained models; simultaneously, the authors highlight that LLMs can perform well on low-resource tasks with transfer learning.

### 2.2.4 Scaling law and capabilities

As the model's scale, they unlock new capabilities not present in smaller models (65). These capabilities are in-context learning(introduced by GPT-3 (38)), instruction following (66), and Chain-of-Thought reasoning (67). There are two aspects of scaling, as discussed below. The first aspect researchers have addressed is the KM scaling law (68) in which, as models, data, or computing resources scale, the model significantly improves its capacity since it depends on these factors. Another aspect is the one used with the Chinchilla model, which requires that the model and data should scale in tandem, while in the KM scaling law, model scaling is preferred more than data scaling. The laws of scaling are important because they have assisted research on understanding LLMs. For example, predictable scaling (55) is used in experimental settings by testing the behavior of smaller models and predicting that it will work better on the bigger variant. Another example is task-level predictability, which indicates a complex link between language modeling loss reduction and actual performance on tasks. Although lower modeling loss suggests the potential for better task outcomes, this doesn't always translate to improved performance, with some tasks even showing inverse scaling, meaning worse performance (69).

### 2.2.5 Pre-training LLMs

To pre-train large language models, there are various corpora which are commonly used from many domains, including Books, CommonCrawl (70), Reddit Links (71), Wikipedia (72), coding datasets[3][4] (73) as well as web-scraped data, which is often not documented or curated. After pre-training, models undergo further refinements like Instruction tuning and Alignment tuning (66) using specific NLP, daily chat, and alignment datasets (51). It is crucial to highlight the importance of high-quality, diverse data sources and careful data preparation to avoid issues like data duplication, which can impact model performance. Data scheduling is also crucial, where the mix and the order of data presented during training can significantly affect the model's capabilities and generalization (51).

### 2.2.6 Tuning strategies

**Instruction tuning**: Instruction tuning (IT) is an advanced technique developed to enhance the capabilities and controllability of large language models (LLMs). This method involves finetuning LLMs on datasets consisting of (INSTRUCTION, OUTPUT) pairs, thus bridging the gap between the models' next-word prediction objective and the user's objective of having models adhere to human instructions (66). Traditionally, LLMs like GPT-3 (38), PaLM (45), and LLaMA (48) have been trained to minimize contextual word prediction error on large corpora, which often leads to a mismatch between the models' training objectives and users' needs for the models to follow instructions accurately (74; 38; 75; 59; 62). IT addresses this gap by further training LLMs on datasets comprising (INSTRUCTION, OUTPUT) pairs, thereby transforming the model's behavior to align with specific user instructions (76; 77).

The general pipeline of IT involves constructing an instruction dataset, either from annotated natural language datasets or generated using LLMs themselves, to finetune pre-trained models in a fully supervised manner, where the model learns to predict each token in the output sequentially given the instruction and input (78). This process not only improves the models' adherence to human instructions but also enhances their predictability and controllability, making them more useful and safer in practical applications (79; 80; 81).

Instruction datasets are pivotal in IT and can be human-crafted, derived from synthetic data via distillation, or generated through self-improvement

[3]https://github.com/
[4]https://stackoverflow.com/

techniques. Prominent examples include Natural Instructions, InstructWild, and Self-Instruct (82). Several notable models have emerged from IT, such as InstructGPT, which shows improved performance in truthfulness and reduced toxicity, and models like BLOOMZ (82) and WizardLM (83), which are finetuned on extensive multilingual and instruction datasets, enhancing zero-shot and multi-modal task performance (66).

Despite its benefits, instruction tuning faces challenges such as creating high-quality instruction datasets and the potential for models to learn surface-level patterns instead of truly understanding tasks (84). These issues highlight the need for continued research and development to refine instruction tuning methodologies and maximize the potential of LLMs (85; 86).

**Alignment tuning**: Moreover, the LLMs must be aligned with human values through a process called alignment tuning (66). Alignment tuning adjusts LLM behaviors to human values, considering criteria like *helpfulness*, *honesty*, and *harmlessness*. Since the training data can often be high or low quality, it may produce toxic content when generating text. To address this issue, the models undergo an alignment process with human values, which takes place during the finetuning of the model.

For this process to be successful, high-quality human feedback is crucial for alignment, a process called reinforcement learning with human feedback (RLHF) (66; 87). RLHF involves several key steps:

**Supervised Finetuning**: Initially, the language model is finetuned on a supervised dataset containing input prompts and desired outputs. Human labelers create these prompts and outputs to cover a diverse set of tasks (66).

**Reward Model Training**: The reward model is trained using human feedback data where the language model generates outputs for various prompts, and human labelers rank these outputs. The reward model is trained to predict these rankings, learning to reflect human preferences (66).

**Reinforcement Learning Finetuning**: The language model is then finetuned using a reinforcement learning algorithm, such as Proximal Policy Optimization (PPO). The model generates text based on prompts, and the reward model provides reward signals that guide the learning process. A penalty term is included to prevent the model from deviating too much from its pre-trained behavior (66).

However, aligning the model with human values can often lead to a loss in model performance, known as the *alignment tax*. Reported performance losses include reductions in creativity and the ability to generate diverse outputs. For example, models trained with RLHF might generate less harmful responses but become overly cautious, limiting their overall utility and informativeness (51; 66). Additionally, the RLHF process can be resourceintensive and complex, involving simultaneously training multiple models and sensitivity to hyperparameter settings (66).

### 2.2.7 Prompting strategies

Other aspects of tuning LMs are done through prompting strategies and in-context learning (38; 88).

In-context learning (ICL) is a significant feature of LLMs, which emerged with the release of GPT-3. This ability allows the model to perform tasks based on examples or instructions within the input text without further training or updating the model's parameters. The concept of in-context learning is critical because it showcases the adaptability and flexibility of LLMs in handling a wide range of tasks by leveraging the context provided in the prompt (38).

Another ability of LLMs is Chain-of-Thought reasoning (67). This ability enables the LLMs to use the prompting mechanism to solve complex problems by following a chain of thought (intermediate steps) to reach the solution. This ability emerges likely when training with code data. Arithmetic reasoning performance is boosted in the PaLM (45) and LaMDA (62) models when the models exceed the 60B parameters.

## 2.3 Multilingual Models

In the realm of natural language generation (NLG), having the capability to interpret and generate into other languages is a vital task. This translation assists in a more inclusive and globally connected digital ecosystem. Recent advancements have emerged with the Transformer models debut (35). These models, designed to understand and generate text across multiple languages, are revolutionizing how we approach language technology. They break the barriers of language-specific systems, offering a unified framework that caters to a diverse linguistic landscape. The significance of multilingual models lies in their ability to leverage cross-lingual similarities and shared knowledge. These models enhance performance in high-resource languages and extend state-of-the-art technology to low-resource languages, which have traditionally been underrepresented and will also be a focus of this paper.

Below, we discuss some popular and state-of-the-art multilingual models, and then we look at how machine translation can improve model performance in under-resourced languages.

Kale et al. (89) introduced mT5, a multilingual variant of the T5 model, pre-trained on a new Common Crawl-based dataset covering 101 languages.

The authors focused on developing a model that inherits the advantages of T5 while expanding its capabilities to multiple languages. The paper details mT5's design, training, and performance on various benchmarks. It also addresses the "accidental translation" challenge in zero-shot settings where a generative model chooses to translate its prediction into the wrong language. It mitigates the issues by proposing mixing unlabeled pre-training data during finetuning. The results demonstrate mT5's state-of-the-art performance in multilingual tasks. The paper contributes significantly to natural language processing by offering a versatile and powerful tool for multilingual text-processing tasks.

Liu et al. (90) present mBART, which is the multilingual version of BART (25), a denoising auto-encoder pre-trained on large-scale monolingual corpora across 25 languages using the BART objective. It is the first to pre-train a full sequence-to-sequence model by denoising full texts in multiple languages and can be finetuned for various translation tasks. The architecture is based on a standard sequence-to-sequence Transformer with 12 encoder and decoder layers. Key findings include substantial BLEU score improvements for low-resource machine translation and document-level unsupervised models. The authors found that pre-training in more languages is particularly beneficial when the target language has limited monolingual data. Interestingly, pre-training on diverse languages does not require significant vocabulary overlap to improve translation quality, suggesting that the model learns universal language properties. Future work may explore finetuning unseen languages on the source side and multilingual pre-training for multilingual machine translation.

Lastly, Moussallem et al. developed a Graph-based RDF Neural Verbaliser named NABU (91), a neural model for converting RDF data into multilingual text. NABU uses an encoder-decoder architecture with a graph attention network encoder and a transformer decoder. It is designed to work with knowledge graphs, which are inherently language-agnostic. The model has been shown to outperform existing methods in English and performs consistently across multiple languages, demonstrating the utility of multilingual models in language generation tasks. The approach suggests future exploration into various graph neural architectures and methodologies to enhance NABU's performance further, particularly in low-resource scenarios and across different knowledge graphs.

### 2.3.1 Machine Translation aid

There are many occasions when the multilingual models do not suffice, and researchers have aided their models with data augmentation techniques or pivoted through English and then to the under-resourced language. These techniques aim to boost the generating process of under-resourced languages by providing additional data that aids the model. Notable approaches are those of Lorendi and Belz (92) and of Kumar et al. (93). As described in more detail in section 2.4.3, the first used the GPT-3.5 and 4 models to few-shot into English and then translate into the under-resourced language achieved the best results in the WebNLG 2023 challenge. The latter used the data augmentation approach and better translation systems, especially for low-resourced languages, to create parallel data of higher quality than the one provided.

Agarwal et al. (94) explore a bilingual system for converting structured data into text and vice versa, improved by machine translation. Based on the T5 model, the system shows significant gains in English and Russian text generation and parsing tasks by incorporating machine translation in pre-training and finetuning phases, particularly enhancing performance on previously unseen data and the Russian language. The research emphasizes the benefit of combining machine translation with bilingual multitasking to improve data-to-text systems for languages with limited resources.

## 2.4 WebNLG best-performing models

We cannot ignore the impact of the chosen model without considering the top-performing models on the WebNLG challenges. More specifically, the WebNLG corpus is a benchmark dataset used in the WebNLG challenge created by Gardent et al. (2). The participants are tasked to submit their proposals, which automatically convert non-linguistic data from the Semantic Web into a textual output (28). Initially focused only on English, the challenge evolved to include new languages, giving another dimension to the challenge by promoting multilingual models. The corpus is discussed more in section 3.1.1. We have studied the top 3 models based on the leaderboard tables, which are usually a modified version of T5 or BART. Since our problem involves machine translation, we will discard the base versions of the T5 and BART and use their multilingual variants, such as mT5 and mBART. The following subsections briefly describe the top emerging models dating back to the 2017 WebNLG challenge. As our primary focus is data-to-text generation using the WebNLG corpus, we must recognize the contribution of previous research with the corpus since it can help us achieve the objectives of this paper.

### 2.4.1 WebNLG 2017

The WebNLG 2017 challenge (28) has the mapping of data-to-text in English as a primary task, as seen in figure 4. The data provided is of Data/Text pairs where the data is a set of triples extracted from DBpedia, and the text is a verbalization of these triples. The task involves specific NLG subtasks such as sentence segmentation (how to chunk the input data into sentences), lexicalization (of the DBpedia properties), aggregation (how to avoid repetitions), and surface realization (how to build a syntactically correct and natural sounding text)

The UPF-FORGe Model (95) is a pipeline system notable for its innovative approach, utilizing linguistic predicate-argument structures to process DBpedia properties. This method transforms RDF triples into these structures, which are then used in a multi-stage sentence generation process. This process includes the careful population of templates, meticulous sentence planning, and sophisticated linguistic generation, integrating rule-based graph transducers with a statistical component for linearization. The model scored first in the METEOR evaluation metric and fourth in BLEU and TER.

The Melbourne Model (96) is an NMT system that adopted an encoder-decoder framework reminiscent of machine translation systems. It focuses on processing RDF triples and their translation into natural language. Key stages include determining the types of entities involved, creating tailored training data, and a strategic de-lexicalization process for entities within sentences. This model scored first in all evaluation metrics of the 2017 WebNLG challenge.

The PKUWriter model (97) is a sophisticated approach to natural language generation. It leverages OpenNMT, a finetuned sequence-to-sequence (seq2seq) model, and TensorFlow's seq2seq API, incorporating features such as bidirectional encoders, Luong attention, and dropout layers. Additionally, the model integrates reinforcement learning to enhance its performance, ensuring that subjects from the input triple set are accurately described in the generated text. This integration leads to a notable improvement in BLEU scores. The model also employs a unique Learning-to-Rank technique to select the best output from multiple-generation models, further enhancing its effectiveness. This comprehensive approach allows PKUWriter to achieve good results, giving it the third position in BLEU and TER evaluation metrics.

### 2.4.2 WebNLG 2020 models

The WebNLG 2020 challenge (98) focused on four tasks: RDF-to-English, RDF-to-Russian, English-to-RDF and Russian-to-RDF. The landscape for the RDF-to-text generation has not changed much since the 2017 challenge, but the Russian language has been introduced. Participants were tasked with generating coherent text from sets of RDF triples. Meanwhile, Text-to-RDF Semantic Parsing introduces a new task to the challenge. This new task required converting text into the corresponding set of RDF triples. It involved reverse-engineering the RDF-to-Text Generation task, effectively translating natural language into structured RDF data.

The challenge aimed to expand the datasets, cover more semantic categories from DBpedia, include an additional language, and promote the development of knowledge extraction tools.

The FBConvAI team (99) used BART as their choice model. They first experimented with BART Base to find the best configuration and deployed the best setting on BART Large, utilizing the model to bridge the gap for the KG-to-text task. The authors introduce methods to improve the model's structural awareness by organizing input and multitask learning for optimal ordering. They also bridge the domain gap between pre-existing text-to-text models and graph-to-text tasks by second-phase pre-training on similar datasets and extra lexicalization to make the generated output more similar to natural language. The efficacy of these methods is demonstrated on the WebNLG dataset on the D2T task in English and Russian, achieving good results regarding text structure and fluency in English but saw a drop in performance in terms of data coverage compared to the T5 models. Overall, FBConvAI was one of the top performers.

OSU Neural NLG (100) uses two models to improve text generation for unseen categories in English and Russian. With limited training data, the option for a pre-trained model approach emerged, explicitly using T5 for English and mBART for Russian. They found that models trained from scratch performed poorly, especially on unseen categories. The models were finetuned with specific hyperparameters to optimize performance. They used BLEU-4 and BLEURT for evaluation, noting that T5 performed better but could not improve further, possibly due to nearing ceiling correctness. The study analyzed model successes and failures qualitatively by comparing training data proportions. One issue of OSU Neural NLG is the hallucinations and omissions, often occurring due to uneven data distribution. Despite some challenges, the pre-trained models showed remarkable results, particularly the T5 in English. The model participated only in the D2T tasks, being one of the top-performing models overall.

AmazonAI (101) came with a novel approach, P2, developed for the WebNLG 2020 Challenge. Their approach focused on converting knowledge graphs to text, utilizing a Relational Graph Convolutional Network (R-GCN) planner and the pre-trained T5 sequence-to-sequence model. The R-GCN planner organizes knowledge graph triplets in an optimal order for text generation, which T5 verbalizes. This method showed superior performance, securing first place in automatic and human evaluations for the English RDF-to-text generation task and in unseen entities and categories.

An honorable mention to the challenge was the bT5 model (94). It is a bilingual Data-to-Text Generation and Semantic Parsing system that leverages a text-to-text generator to handle tasks in both English and Russian, exploring multilingual multi-task learning during pre-training and finetuning. Machine translation aids the model during both pre-training and finetuning stages. Lastly, the bT5 model was the only team that participated in all four tasks of the WebNLG 2020 challenge, and overall, it achieved good results amongst the top-performing ones in every task.

### 2.4.3 WebNLG 2023 models

The WebNLG Challenge 2023 (102) was organized to address the gap in text generation research for Russian and under-resourced languages, focusing on Breton, Irish, Maltese, and Welsh. The challenge involved converting RDF triples into natural text, emphasizing these less-represented languages. The contest attracted a variety of approaches, including rule-based and machine-learning-based systems. The systems were evaluated automatically and by human judges, considering criteria like fluency, grammaticality, and factual accuracy. The results demonstrated varying levels of success across languages and approaches, highlighting the challenges and potential strategies for text generation in under-resourced languages.

The Cuni-Wue model (93) was one of the participating models that tackled all four tasks provided by WebNLG 2023. They chose an mT5 model focusing on multilingual RDF-to-text generation. They used MT primarily for data augmentation and quality improvement in multilingual RDF-to-text generation, especially targeting low-resource languages. They aim to create or enhance training datasets by translating existing resources into target languages. For languages like Maltese (Mt), Irish (Ga), and Welsh (Cy), they use the NLLB system for translation, replacing the Edinburgh Zero System. For Breton (Br), where NLLB is unavailable, they filter out inconsistent examples from the training data. At the same time, they focus on multitask learning, which means simultaneously training models on diverse tasks to improve generalization. The approach combines data-to-text generation in the target language with translation from English and data-to-text in English. Another technique used is the split and generate(SaG), which addresses the issue of data quality in larger input triple sets; they adopt a method of splitting complex inputs for easier processing. The input triple sets are split into subsets, and outputs are generated for these subsets individually before being concatenated. Their experiments showed substantial improvements in language generation for low-resource languages using these methods. For example, they observe notable improvements for Mt, Ga, and Cy using the NLLB system. However, like Russian (Ru), the SaG decoding method sometimes reduces performance due to the language's comparative resource richness. The model achieved good results, occupying the second place in the contest leaderboard most of the time.

The WebNLG-Interno model (103) was only used for the Russian RDF-to-text generation task in the WebNLG Challenge 2023. The authors used FRED-T5, a pre-trained LLM, and experimented with different prompts to enhance its performance. They also used some preprocessing steps mentioned by Agarwal et al. (94), achieving a Translation Edit Rate (TER) of 0.373 on the test dataset, outperforming the best result of the previous challenge. Their approach included using additional information from raw XML data and machine translation. One of their findings is that even though translated data can be beneficial, it is not crucial for significant improvements.

The DCU-NLG-PBN model (104) focused on using LLMs like GPT-3.5 and GPT-4 for data-to-text generation tasks in under-resourced languages. These languages include Irish, Maltese, Welsh, and Breton, but the model is not finetuned on WebNLG data. The study investigates how LLMs, which are predominantly trained on English data, perform in generating text for languages with limited representation in their training data. This investigation aligns with the work of Brown et al. (38), where LLMs use their broad knowledge to adapt to new tasks at inference time, given a few demonstrations for the task at hand. In-context learning allows the model to perform tasks based on the instructions without any updates to its parameters, in the case of the DCU-NLG-PBN generating on under-resourced languages in which the training data might contain very few samples or even none. Their research was two-fold: direct generation in the under-resourced language and generation in English followed by translation into the target language. They found that few-shot prompting was more effective for direct generation, but this advantage disappeared when using English as a pivot language. Combining few-shot prompting with translation (using Google Translate) produced significant improvements, outperforming competitor systems in the WebNLG 2023 shared task across all languages on various metrics. However, the performance in Welsh, the best among the tested languages, still lagged behind the lowest-ranked English system from WebNLG'20. Moreover, the reproducibility of the experiments is in question since they don't have access to the model but only use some available endpoints provided by OpenAI. Overall, one of the important papers shows the power of LLMs since it outperformed all other models in the challenge in the languages it was tested on. A simple setup can produce a good model for under-resourced language generation.

## 2.5 Evaluation

To assess the quality of the text generated from RDF triples, we will utilize automated evaluation metrics provided by WebNLG, including BLEU, METEOR, chrF++, TER, and BERTScore. Unfortunately, the BERT model for the Maltese language does not support BERTScore, so it will not be utilized. Additionally, ROUGE and PARENT will be employed, and we may seek the input of human evaluators to assess the generated text based on four distinct criteria.

### 2.5.1 BLEU (Bilingual Evaluation Understudy)

The BLEU metric (19) is one of the most popular and inexpensive algorithms for evaluating the quality of text that has been machine-translated from one language to another. The main idea behind BLEU is that it is better if the correspondence between a machine translated and the human reference is close.

Scores are calculated for individual translated segments, generally n-grams, by comparing them with good-quality reference translations. Those scores are then averaged over the whole corpus to estimate the translation's overall quality. Neither intelligibility nor grammatical correctness are taken into account. Even though it is one of the first metrics to report a high correlation with human judgments, the model suffers from many limitations. Firstly, since BLEU is an n-gram-based metric, it compares token overlap from the predictions and references instead of comparing meaning. This can lead to discrepancies between BLEU scores and human ratings. Secondly, shorter translations achieve higher scores than longer ones simply due to how the score is calculated. To counteract this, a brevity penalty is introduced. Thirdly, BLEU scores are inconsistent when comparing different datasets or languages. Lastly, scores vary greatly depending on which parameters are used to generate the scores, especially when different tokenization and normalization techniques are used. Therefore, it is impossible to compare BLEU scores generated using different parameters or when these parameters are unknown. For the experimental setup, the BLEU score is just an easy metric to calculate, and it might provide the linguistic and phrasal accuracy of the generated text, given the reference text.

However, this metric can be problematic since there have been many implementations of it. Post (105) highlights the inconsistent reporting of BLEU scores within the machine translation research field. He mentions how BLEU, a variable metric, yields diverse scores due to different, often undisclosed, parameter settings. The varying tokenization and normalization methods on these scores complicate comparisons across studies. For this reason, we will use a uniform BLEU standard, as used by the Conference on Machine Translation (WMT), called SACREBLEU, a tool to ensure consistent, transparent BLEU reporting, ensuring comparability with other research studies.

### 2.5.2 METEOR (Metric for Evaluation of Translation with Explicit Ordering)

The METEOR evaluation metric (20) is another automatic evaluation metric based on unigram matching between the machine-produced translation and human-produced reference translations. Unigrams can be matched based on their surface forms, stemmed forms, and meanings, providing a more nuanced understanding of meaning in translations. Once all generalized unigram matches between the two strings have been found, METEOR computes a score for this matching, using a combination of unigram-precision, unigram-recall, and fragmentation that is designed to directly capture how well-ordered the matched words in the machine translation are in relation to the reference.

For our evaluation purposes, the metric also has synonym and paraphrase limitations and morphological complexity. Since there are big chances for data augmentation using automated translation tools, the metric might not do justice when using the automatically translated text to assess the quality of the generation. Also, in languages with different morphological complexity, mapping words based on unigrams might do poorly since there are languages that express the same meaning differently with more or less unigrams. The metric requires more testing for human correlation with other languages.

### 2.5.3 chrF++(Character-level F-score)

The chrF metric (106) is a method for evaluating machine translation output using character n-gram F-scores. This metric is advantageous because it captures some morpho-syntactic phenomena and is language-independent and tokenization-independent. It doesn't require additional tools or linguistic resources, making it broadly applicable. The chrF score is computed by calculating character n-gram precision and recall and then combining these using the F-score formula. It has shown promising results in correlating with human judgments of translation quality. The chrF++ metric (107) extends over chrF by incorporating character n-grams and word n-grams (specifically, unigrams and bigrams) to evaluate machine translation outputs. It correlates better with human judgment than metrics focusing solely on character or word n-grams, and it is effective in assessing translations of morphologically rich languages. In our scenario, chrF++ can provide a more comprehensive evaluation of the translation quality, especially considering the morphological richness of these languages. This could highlight model performance in preserving the meaning and structure of the source data in diverse linguistic contexts.

### 2.5.4 TER (Translation Edit Rate)

The translation error rate (or translation edit rate) (22) explicitly tries to estimate the amount of work required to turn the machine translation(MT) output into the reference translation.

Specifically, it quantifies the number of edit operations (insert, delete, substitute, shift) required to change the MT output into the reference translation. This could help with detailed error analysis. Moreover, TER is generally language-independent and can be applied to various language pairs. On the other hand, it lacks semantic understanding since TER ignores any notion of semantic equivalence. A translation that semantically matches a reference translation needs human knowledge. Additionally, TER might not adequately represent fluency or naturalness in translation, focusing more on lexical and syntactic accuracy. For TER, the quality of the reference translation significantly influences TER scores. Poor-quality references can lead to misleading results, but the WebNLG corpus will not suffer from this issue since professional translators have manually handcrafted all translations so that this metric can give pretty good insights into our system generation quality.

### 2.5.5 BERTScore

BERTScore (24) is an automatic evaluation task-agnostic metric for text generation tasks that correlates well with human judgments. The simple principle behind BERTScore is to calculate a cosine similarity score for each token in the candidate sentence and each token in the reference sentence. Instead of capturing n-gram overlap, it computes token similarity using contextual embeddings using the BERT model (37). This enables the model to capture semantic equivalences between generated and reference texts, particularly useful in NLG tasks since the generated output might be correct but paraphrased. Another benefit of this evaluation metric is that since it has been trained in diverse languages, it can be used to evaluate multilingual models. The metric's performance relies heavily on the quality of BERT's contextual embeddings. However, the model might suffer unstable performance against low-resourced languages, thus affecting its reliability in the WebNLG setting. Additionally, the BERTScore does not support Maltese. We will have to use other evaluation metrics to assess it. Lastly, BERTScore computes BERT_Precision, BERT_Recall, and BERT F1, which can be used to evaluate different language tasks.

### 2.5.6 ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

The ROUGE metric (108) measures the overlap between a generated summary and reference summaries using various methods like n-gram overlap (ROUGE-N), longest common subsequence (ROUGE-L), and skip-bigram co-occurrence statistics (ROUGE-S). ROUGE can evaluate different aspects of summarization, like content overlap and word order, and its adaptability to different summary lengths. However, its reliance on reference summaries and potential limitations in capturing semantic variations are drawbacks. Since data-to-text generation involves creating descriptive or informative text from structured data, ROUGE can be helpful for our project since it is a cheap metric. The metric could be useful for evaluating how well the multilingual model generates text in Irish, Maltese, Breton, and Welsh and if it captures the key information from structured data sources, particularly regarding content overlap and information retention.

### 2.5.7 PARENT(Precision And Recall of Entailed N-grams from the Table)

The PARENT metric (109) evaluates table-to-text generation models but could also be applied for RDF-to-text generation. It assesses the quality of generated text by comparing it with reference texts and the source table. It calculates precision and recall by aligning n-grams from the generated and reference texts to the table data, using an entailment model to determine if the table entails a text n-gram. This method helps address issues arising from divergent reference texts that may contain information not present in the table, a common scenario in automatically constructed datasets.

In the context of WebNLG, PARENT is instrumental since it correlates highly with human judgments. The WebNLG dataset presents structured data (RDF triples) and corresponding text descriptions. PARENT evaluates how well the generated text aligns with the factual content of these triples, meaning that it will assess the model's ability to convert data into coherent and factually correct text in the target languages. PARENT evaluates the fidelity of the generated text to the source data.

When PARENT generates text from tables, it doesn't use the complete tabular data. This can be an issue as in the WebNLG corpus since the aim is to generate all the information in the RDF triples.

# 3 Methodology

In this section, we discuss the methodology for achieving our goals. We describe the datasets we will use and then discuss the experimental setup, including tools, data augmentation techniques, data preprocessing, the proposed models, and the pipeline we implement. Then, we discuss the task description.

## 3.1 Data

This section presents the datasets we will use in our pipeline to test zero-shot generation on low-resourced languages. The model will primarily use the OPUS-100 dataset for additional pre-training and the WebNLG for finetuning.

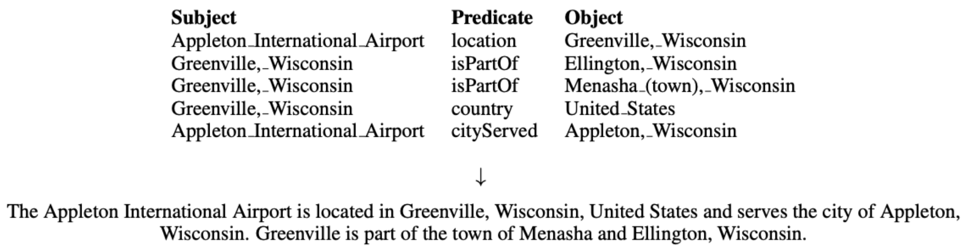
### 3.1.1 WebNLG corpus

The WebNLG dataset (28) contains *Resource Description Framework* (RDF) triples extracted from DBPedia, and the text is a verbalization of this data. Each triple is in the form of *Subject, Predicate* and *Object.* The *Predicate* represents the relationship of the *Subject* and *Object.*

The WebNLG corpus was initially developed for English, consisting of 25,298 texts describing 9,674 sets of up to 7 RDF triples in 15 domains: *Astronaut, University, Monument, Building, Comics Character, Food, Airport, Sports Team, Written Work, and City.* New domains were added later to create the test set. These domains are *Athlete, Artist, Means of Transportation, Celestial Body* and *Politician* were used for the test set.

The last version of the WebNLG corpus released in 2023 targets a variety of under-resourced languages, such as Breton, Welsh, Irish, and Maltese. The latest dataset contains 1,399 dev items for Breton, 1,665 dev items for Welsh, Irish, and Maltese, and 790 for Russian. WebNLG also provided 5,573 train items for Russian and 13,211 for the other languages. These items were automatically translated using an NMT system (110). Lastly, the corpus offers 1,101 test items for Russian and 1,778 for other under-resourced languages.

The challenge uses various metrics to evaluate the quality of the proposed solutions, including traditional NLG metrics like BLEU (19), METEOR (20), chrF++ (107), TER (22), and BERTScore (24), as well as human evaluations for fluency, adequacy, and correctness. Figure 4 shows an example of 5 RDF triples verbalized to English text.



| **Subject** | **Predicate** | **Object** |
|---|---|---|
| Appleton_International_Airport | location | Greenville,_Wisconsin |
| Greenville,_Wisconsin | isPartOf | Ellington,_Wisconsin |
| Greenville,_Wisconsin | isPartOf | Menasha_(town),_Wisconsin |
| Greenville,_Wisconsin | country | United_States |
| Appleton_International_Airport | cityServed | Appleton,_Wisconsin |

↓

The Appleton International Airport is located in Greenville, Wisconsin, United States and serves the city of Appleton, Wisconsin. Greenville is part of the town of Menasha and Ellington, Wisconsin.

***Figure 4.*** *RDF-to-text verbalisation*

### 3.1.2 OPUS-100

The development of the OPUS-100 dataset (110) has emerged due to the recent advancements in multilingual NMT. This English-centric dataset, sourced from the OPUS collection, includes 100 languages with up to 1M training pairs for each language pair, totaling approximately 55M sentence pairs. The dataset composition varies, from movie subtitles to GNOME documentation and the bible. For training, validation, and testing, up to 1M, 2,000, and 2,000 sentence pairs were randomly sampled per language pair, respectively, with careful filtering to prevent overlap. Additionally, OPUS-100 serves zero-shot translation evaluation in 15 language combinations like Arabic, Chinese, and French, further underscoring its broad applicability in the NMT domain.

## 3.2 Experimental setup

This section presents the experimental setup we will use for the project implementation. In the following sections, we describe the languages we will use, data augmentation techniques and accompanying tools, the data preprocessing stage, and the decisions that led to choosing the right model for the task. Then, we give an overview of the systems pipeline describing each stage, and lastly, we present the task this model will address.

### 3.2.1 System overview

This section explains the modeling choices for each step of the pipeline to generate text from RDF triples. We aim to explain each step of the pipeline to enhance the transparency and replicability of our research. The proposed pipeline is illustrated in figure 5.

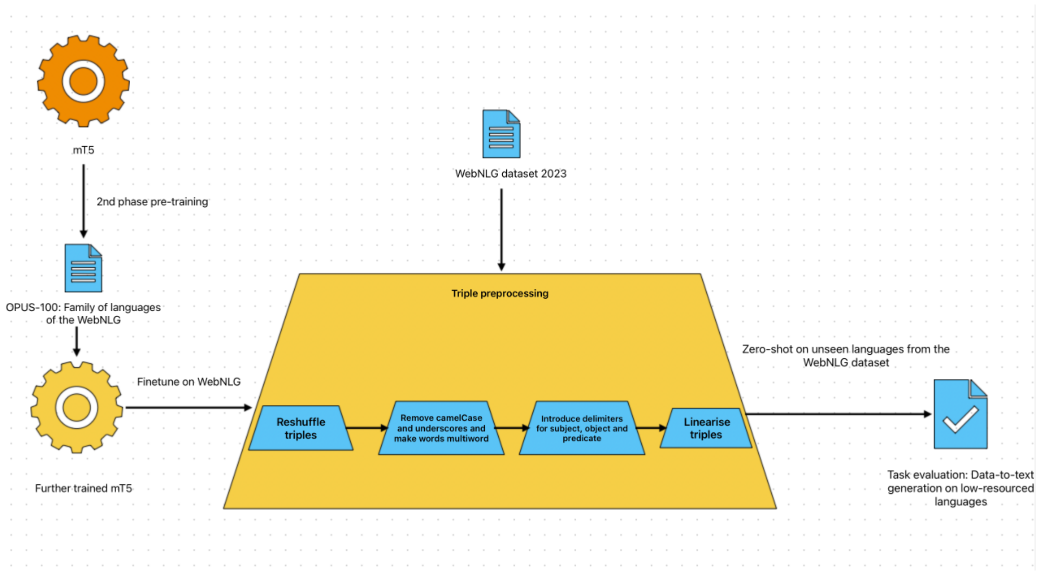

mT5
2nd phase pre-training
OPUS-100: Family of languages of the WebNLG
Finetune on WebNLG
Further trained mT5
WebNLG dataset 2023
Triple preprocessing
Reshuffle triples
Remove camelCase and underscores and make words multiword
Introduce delimiters for subject, object and predicate
Linearise triples
Zero-shot on unseen languages from the WebNLG dataset
Task evaluation: Data-to-text generation on low-resourced languages

***Figure 5.*** *Proposed pipeline for generating data-to-text in low-resourced languages*

To commence, we used a multilingual model, specifically mT5 (89), which is detailed in section 3.2.5. While mT5 is already trained in various languages (see Appendix B), we pre-train the model using the OPUS-100 dataset. This step further enriches the model's general knowledge with all the relevant languages we want to target in the WebNLG dataset. We exclude the target languages in both corpora to test if the language family alone can generate a decent result in a zero-shot setting, but we include Irish in the pre-training because there are not many languages that are closely related to Breton. It is important to note that the mT5 model has already been trained with many languages, including the ones from WebNLG. Furthermore, training on the OPUS-100 dataset (110) is straightforward and efficient since it's in text format, eliminating the need to aggregate and extract data ourselves. We leverage the bilingual sets of languages provided in the OPUS-100 dataset. This involves selecting language pairs closely related to the target languages in the WebNLG dataset. The input to the model would be text in one language, while the output would be its translation in another language. This method will allow the model to learn cross-linguistic representations and improve its ability to understand and generate text, which will be useful later for the D2T task.

At this stage, the model has further enriched its linguistic capabilities. Next, we finetune the model to generate text from RDF triples. This process enables the model to learn how to produce fluent text in natural language from structured data. The structured data we will be using comes from the WebNLG corpus (28), and as we described, preprocessing the data makes it easier for the model to understand and train on, as discussed in section 3.2.3. Lastly, we will generate text without providing examples to the model, but we will also experiment with a non-zero-shot setting approach.

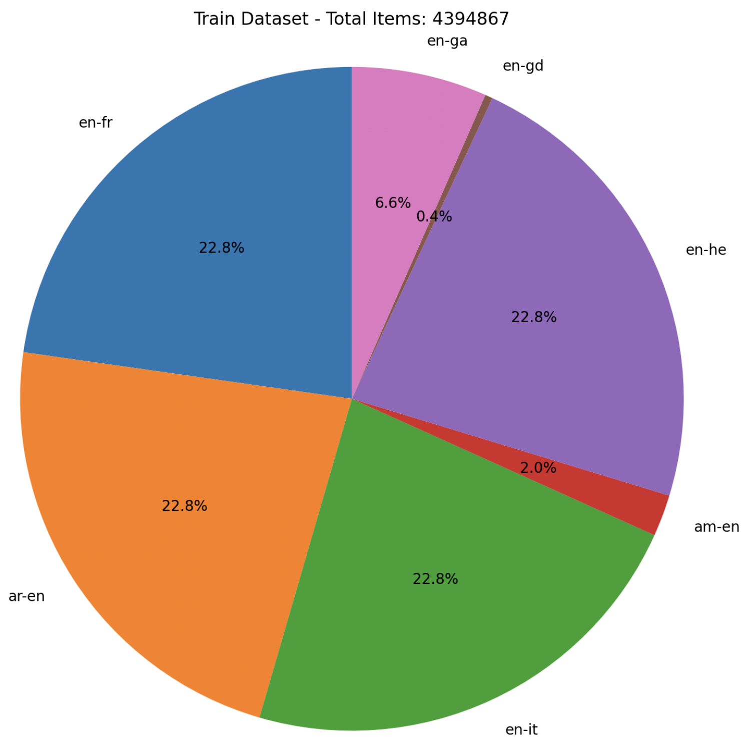
### 3.2.2 Languages

The languages we will use for this project span various language families. We pretrain it in Arabic, Hebrew, Italian, French, English, and Amharic to target Maltese, a Semitic language. We also include language examples from the Celtic languages, such as Irish and Scottish Gaelic, even though Irish is one of the target languages of the WebNLG dataset. We have included examples from Irish and Scottish Gaelic, which are Celtic languages, to target Breton and Welsh, which also belong to the Celtic group. Lastly, to test model performance and robustness, we have a variant of the model finetuned in English, one in English and Russian and another in English, Russian, the OPUS-100 languages, and German. We hope with these diverse results, the model will learn to understand the prompt better and generalize in languages not seen before.

### 3.2.3 OPUS-100 data preprocessing

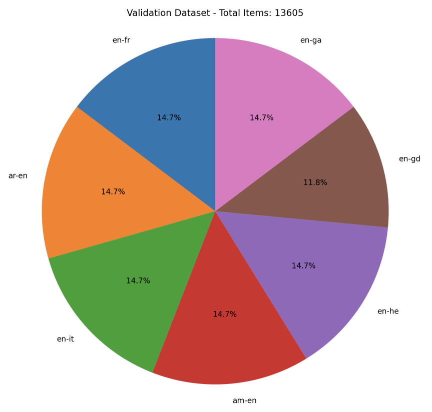
The OPUS dataset (110) language pair distribution is shown in figure 7. For experiments, we use the OPUS-100, a massively multilingual dataset sampled from OPUS (111). OPUS-100 consists of 55M English-centric sentence pairs covering 100 languages (110). The OPUS-100 provided in HuggingFace contains a lot of language pairs that are mistranslated, meaning that the English translation is carried to the target language. This can create a problem in our model's training and performance since the data can be low quality. For this reason, we apply a data preprocessing operation to clean the data from such translation pairs.

Lastly, since the OPUS-100 dataset contains many language pairs in reverse order (e.g., Amharic to English), we process the dataset to reverse the ordering and re-arrange it as English to Amharic. The aim is for the decoder of mT5 to be exposed to these different languages to acquire better language representations and perhaps be better at cross-lingual transferring. The training, validation, and test distributions are illustrated in figures 6a , 7a and 7b

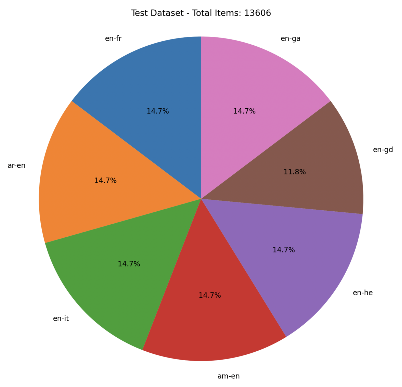


*(a) Original dataset*

***Figure 6.*** *Language pair distributions of the training data on OPUS-100(filtered)*



*(a) Validation dataset*



*(b) Test dataset*

***Figure 7.*** *Language pair distributions of the test and validation data on OPUS-100 (filtered)*

### 3.2.4 WebNLG data preprocessing

Another preprocessing step we must implement is on the RDF triples, especially when they are the input of text-to-text models such as BART (25) or T5 (50). For this reason, it is vital to preprocess the data so that the models can handle its structured nature. This preprocessing step aims to ensure that the data is uniform and consistent, which is crucial for the optimal performance of the model. The ultimate goal of the model is to learn to generate fluent text rather than memorize the ordering of the input data, which can make it less adaptable to other scenarios where the data may be shuffled.

To generate text that accurately reflects the intended meaning of the input, the model should not rely on the order in which the triples are presented. Since linearization can give an unfair advantage to the model, as seen in the work of Moussallem et al. (91), we will implement reshuffling to the triples to ensure the model does not depend on linearization, but instead, accurately represents the intended verbalization order.

To eliminate the camelCase in the data, we will convert them to multiword expressions with normal spacing, as demonstrated in the work of Agarwal et al. (94). For instance, La_Crosse_Wisconsin →La Crosse Wisconsin. This approach will make the data more consistent and closer to natural language. It will also help the tokenizer as case sensitivity may identify subword units differently.

Furthermore, we utilize distinct delimiters to distinguish each triple's

Subject, Object, and Predicate (94). These delimiters, denoted as **<S >**, **<0>**, and **<P>**, serve to enhance the clarity and organization of the data, streamlining the model's ability to comprehend the relationships among the triples. Furthermore, these delimiters are prominent indicators for the model to parse the data effectively.

To finalize the preparation of RDF data, it is necessary to serialize each RDF triple, which involves converting the subject, predicate, and object into a string format. This can be done by concatenating the three parts into a single string.

### 3.2.5 Models

To implement this project, we use pre-trained state-of-the-art transformer-based models with an encoder-decoder architecture. The models of choice are text-to-text models pre-trained on a multilingual corpus. Two models that emerged after research are the T5 (50) and BART (25) models but in their multilingual versions. The mT5 (89) variant of T5 comes in many sizes ranging from small to XXL [5], whereas the mBART (90) can only be found in base size pre-trained on 25 languages and on a large version which is pre-trained on 50 languages [6]. Based on section 2.4, we discussed how the WebNLG challenge has evolved with every iteration and how the participating teams have tried to tackle the challenges. The use of the T5 family is prevalent. Firstly, the model is a text-to-text generator meaning data can be easily transformed into a string. Secondly, for multilingual tasks, the T5 multilingual variant has been trained in many more languages than the BART counterpart(Appendices B C). Thirdly, since the resources are limited for the scope of this MSc thesis, it would be easier to train with a model that is not computationally expensive. As a reference, mT5-small [7] is around $\approx 220M$ compared to $\approx 580M$ parameters for the mT5-base. On the other hand, the BART family has for the mBART[8] variant pre-trained on 25 languages around 680M parameters and >680M parameters for the large.

### 3.2.6 Pre-training

The first involves pre-training our multilingual model by doing one pass of the OPUS-100 corpus with the parameters described in table 1.

[5]https://huggingface.co/google/mt5-base

[6]https://huggingface.co/facebook/mbart-large-50

[7]https://huggingface.co/google/mt5-small

[8]https://github.com/facebookresearch/fairseq/tree/main/examples/mbart

***Table 1.*** *mT5 pre-training setup*

| **Paremeters** | **Values** |
|---|---|
| *num_train_epochs* | 1 |
| *per_device_train_batch_size* | 8 |
| *per_device_eval_batch_size* | 8 |
| *learning_rate* | 0.001 |
| *gradient_accumulation_steps* | 4 |
| *predict_with_generate* | True |
| *optimiser* | AdaFactor |

We provide statistics about the length per language string in figures 8, 9. These statistics help us to set the maximum input length to 128 for the input and target string sequences fed and produced by the model. This process ensures that the strings have a uniform length, simplifying batch processing and improving computational efficiency.

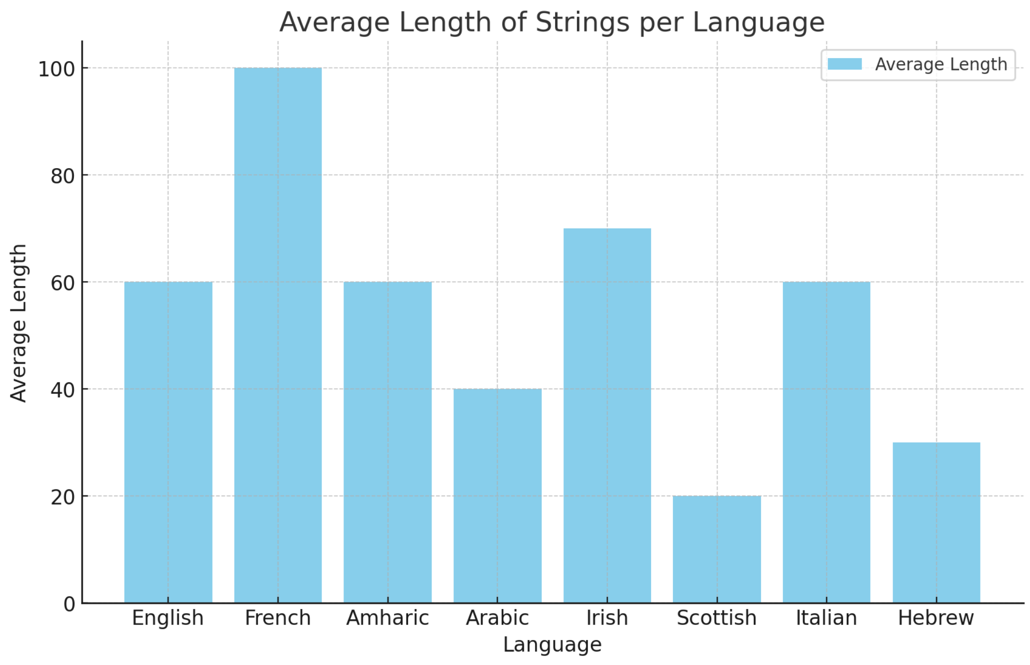


***Figure 8.*** *The average length per language string in the OPUS-100 dataset*

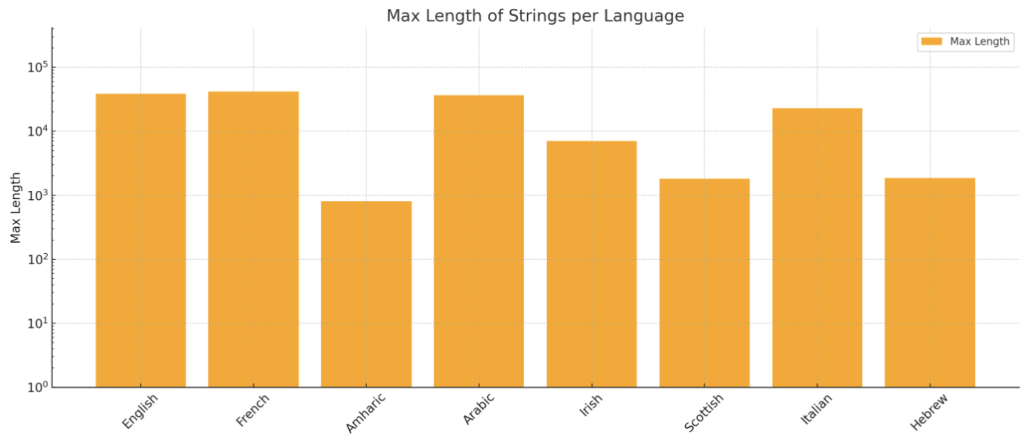


***Figure 9.*** *The max length per language string in the OPUS-100 dataset*

When pre-training the mT5, there are many things that we should take into consideration. For the most successful training regimen, we look over Kaplan et al. (68) work to estimate the ideal scaling of data and model size. Due to hardware constraints, we have chosen the mT5-small model [9] since it contains the least amounts of parameters $\approx 220M$ and it will take less time to pre-train and finetune it. Using the scaling laws for Neural Networks, we want to achieve data scaling so that our model will not overfit and acquire better generalization capabilities.

In the pre-training phase, the model is trained on the mC4 corpus[10] with around 1T tokens (89). Additionally, we try to enrich the models' general knowledge by training it further with the OPUS-100 dataset containing $\approx 5.6 * 10^8$ tokens. To avoid overfitting, we use the stochastic approach of the AdaFactor optimizer (112), which can help prevent the model from seeing the same data multiple times (50). We observe that the validation loss is generally lower than the training loss, but both steadily decrease. The learning curves can be found in figure 20 of the Appendix. The curves' appearance might also be attributed to the model's capacity since we work with the small variant. The model has already been pre-trained for 1M steps on the mC4 dataset (89). Further pre-training might have diminishing returns since the model cannot learn from the new data. Even though the model's curves do not converge and validation loss is generally lower than training loss, we assess the model based on the quality of the generated text and on the improvement of automatic metrics over time. When the improvement plateaus, we stop the model from further training.

[9]https://huggingface.co/google/mt5-small
[10]https://www.tensorflow.org/datasets/catalog/c4#c4multilingual

### 3.2.7 Finetuning

For finetuning, we follow the recipe of the CUNI-Wue model by Kumar et al. (93). The initial plan was to follow the mT5 finetuning strategy presented by Xue et al. (89), but it only provides a general guideline for downstream tasks. For this reason, we adjusted our model parameters based on the CUNI-Wue model.

***Table 2.*** *mT5 finetuning setup*

| Paremeters | Values |
|---|---|
| *num_train_epochs* | 10 |
| *per_device_train_batch_size* | 8 |
| *per_device_eval_batch_size* | 8 |
| *learning_rate* | 2e-5 |
| *gradient_accumulation_steps* | 8 |
| *predict_with_generate* | True |

### 3.2.8 Text generation

We experiment with both zero and non-zero-shot text generation in the WebNLG (102) target languages. We use the settings described in table 3 as a decoding strategy. Initially, our decoding strategy consisted of beam search decode with a beam width of 4 and repetition penalty of 3.5 to prevent the model from repeating the same information (113). These settings did not produce the best quality text since, in many cases, repetition of the same sentence or words was prevalent and hurting performance. We additionally included sampling for randomness for more varied outputs and doubled the beam width to 8 to increase the exploration of multiple possible sequences simultaneously, keeping track of the top 8 most likely sequences at each step. Additionally, we introduced temperature to control the randomness of the predictions and reduce the probability of less likely tokens, leading to more coherent and focused text. The repetition penalty for 2-grams ensured that the generated text did not include repetitive phrases, an issue we faced with the old setup. Last but not least, we enabled early stopping to allow the generation process to halt once a complete and sensible output is achieved rather than continuing to the maximum token limit. The result in the generating text can be seen in figures 17 for the old decoding strategy and figure 18 for the new decoding strategy. For RDF triple verbalization, we use the prompt format: *RDF-to-text in* $< lang >$*:* $< input >$ where $< lang >$ is the target language and $< input >$ denotes the input triples.

***Table 3.*** *Decoding strategy parameters*

| Paremeters | Values |
|---|---|
| *do_sample* | True |
| *max_new_tokens* | 100 |
| *num_beams* | 8 |
| *temperature* | 0.6 |
| *repetition_penalty* | 3.5 |
| *no_repeat_ngram_size* | 2 |
| *early_stopping* | True |

### 3.2.9 Experiments

In the following figure 10, we describe the experiments performed on this project. Moreover, we give a detailed overview of each experiment and the reasoning behind each experiment.

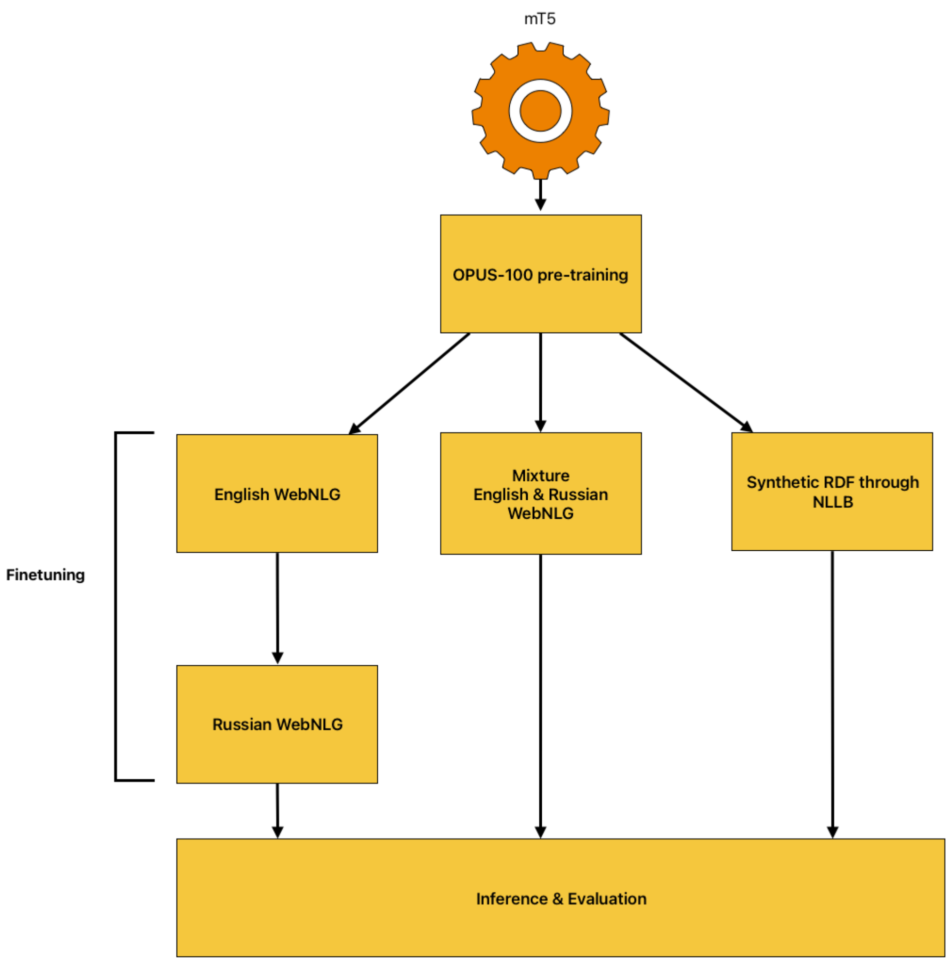


***Figure 10.*** *Proposed experiments for evaluating our model in zero-shot setting*

The chain of experiments measured the impact of each recipe on a zero-shot generation. We started by enriching the model's general knowledge using further pre-training, and then we branched out to different finetuning recipes to determine which yields the best performance on the automatic metrics.

For the first experiment, we performed a single pass on the OPUS-100 since our goal was not for the model to learn the translation task but to familiarise itself with the language families of Irish, Maltese, Breton, and Welsh. Then, we evaluated the model per language pair to analyze the performance on the OPUS-100 translation task. The results for the translation task are found in table 16 and conclude later on how the pre-training might affect the RDF-to-text task. Moreover, we use stopping criteria such as early stopping to avoid overfitting the training data. After the pre-training, we finetune for 100 epochs on the RDF-to-text in English and another 100 on the RDF-to-text in Russian sequentially. Of course, this resulted in the model getting more biased towards Russian and forgetting how to generate in English. To resolve this problem, we kept the checkpoint where the model is finetuned only in English. In the Appendix, we illustrate the figures 19 for the pre-training and 20 for the validation loss curves. Our observations from the first experiment were that the model was mostly generated in English, and it also introduced words from Arabic and Amharic. We speculate that this is due to pre-training. Another speculation was that the model had also experienced catastrophic forgetting by mostly generating in English and not understanding the text instruction.

For the second experiment, we tried to enrich the dataset with more languages, this time by combining Russian and English samples in the same dataset, to improve model robustness and understanding of diverse instructions. We finetuned the model for 200 epochs and then performed zero-shot generation. Our observations from this setup were that the model could still not generate well in zero-shot, but it had seen a significant improvement compared to the first experiment. Following this line of thought, we expected that an even more diversified WebNLG dataset could give the necessary learning capabilities for mT5 to generate even better text. Since WebNLG is limited to a specific set of languages, we created synthesized data using the NLLB model (114).

For the third experiment, we generate an augmented dataset based on the Russian and English WebNLG datasets. We randomly select samples from the data and generate translated lexicalizations based on OPUS-100 languages and by adding German using the NLLB model (114). German was chosen arbitrarily as a language to expand the diversity of the languages the model was exposed to during finetuning. The new dataset consists of 57,795 samples, and the model is trained for 100 epochs. The language distribution can be seen in figure 11. Additionally, for this experiment, we have included the percentage of noisy and clean data from this distribution, illustrated in figure 12. Our observations for this setting were that the model indeed learned to generate and score better, but the diversity of languages present both at the pre-training and at the finetuning stage had caused the model to do code-switching between languages when it was instructed to generate in a specific language.

Lastly, as part of measuring the impact of the pre-training, we finetuned an mT5 model without pre-training by only following the best finetuning strategy from our previously described experiments. Surprisingly, this model performed equally well as its pre-trained counterpart. The performance gains in some aspects might come from the fact that as a multilingual model, it

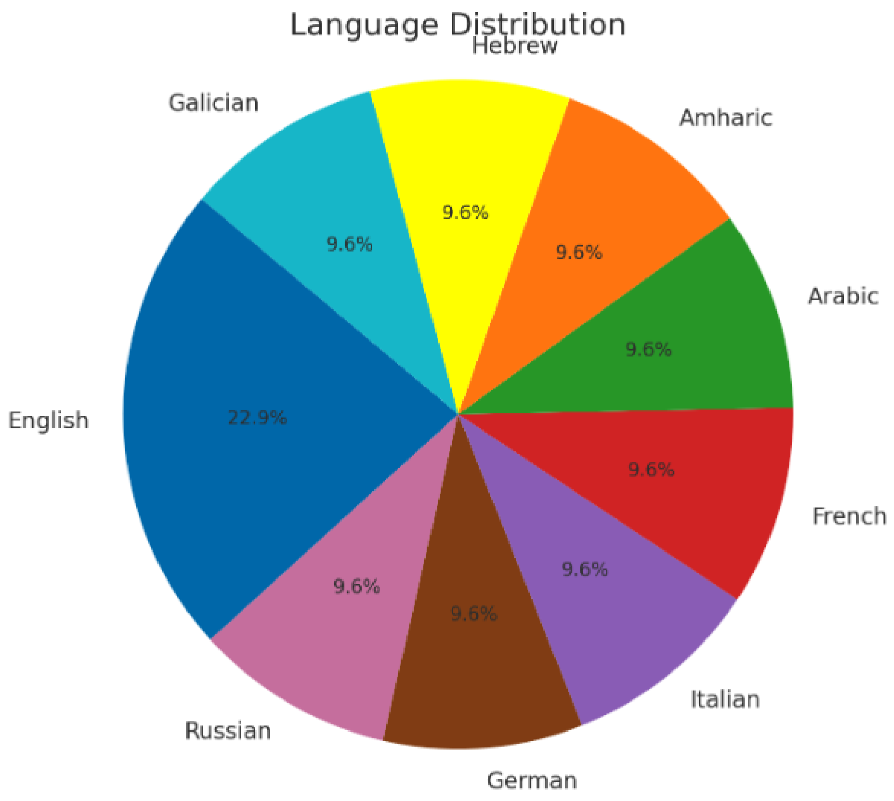


***Figure 11.*** *Language distribution of the synthesized WebNLG*

doesn't need further pre-training, as it is already extensively trained on mC4 [11], making it very knowledgeable out of the box. However, mT5 does require finetuning.

[11]https://www.tensorflow.org/datasets/catalog/c4#c4multilingual

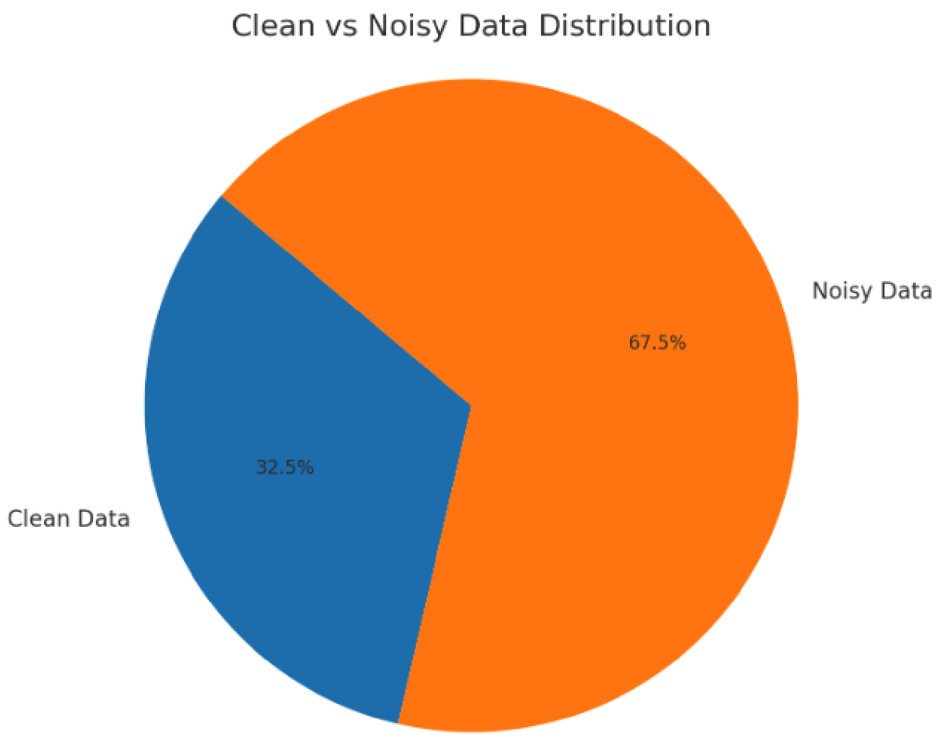


***Figure 12.*** *Distribution of noisy and clean data on the synthesized WebNLG dataset*

### 3.2.10 Second stage finetuning

In this project, we also perform non-zero-shot generation experiments for experimentation. More specifically, we train our model with the target languages from WebNLG after the model has been finetuned to the settings described in the section 3.2.7. The data for training our model is synthesized using the NLLB model (114). We translate pre-existing lexicalizations from English and Russian to Maltese, Irish, and Welsh. We could not synthesize data for Breton since NLLB is not trained in that language.

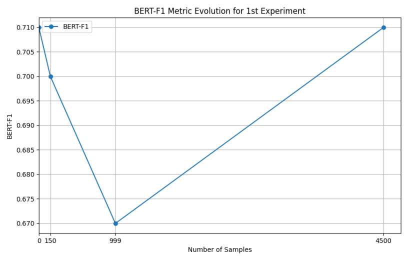
We perform three experiments per the settings described before as a second-stage finetuning approach. The first experiment consists of 150 synthesized samples with 50 samples per language. In the second experiment, we synthesize 999 samples with 333 samples per language. In the last experiment, we synthesize 4500 samples with 1500 samples per language. We selected this size to observe how the performance is affected. We increase the dataset with noisy data and if that can have diminishing returns on the model.

We examined the development of metrics for these experiments, beginning with the BERT-F1 metric per experiment. This is shown in figures 13a, 13b, 13c. The model notation in the figures caption can be found in section 4.

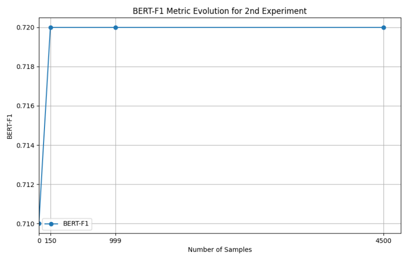
Regarding BERT-F1, the PT_WebNLG_en model initially shows a drop in performance but improves after adding +4500 samples. Conversely, the PT_WebNLG_en_ru model plateaus after just +150 samples, while the PT_Augm_WebNLG model consistently declines as more samples are added.

The remaining figures can be located in the appendix figure in section E. Furthermore, for validation, we utilized the training dataset provided by WebNLG in the Irish, Maltese, and Welsh languages. The training dataset consists of 1998 samples, with each language having 666 samples. Finally, we used the dev dataset provided by WebNLG for evaluation.

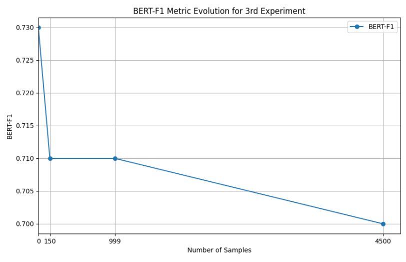
We observed that the PT_WebNLG_en model usually displays an initial performance drop with a small amount of data but significantly improves with larger datasets, exhibiting better generalization with more data. This trend is consistent across metrics such as BLEU, chrF++, and METEOR. The PT_WebNLG_en_ru model shows a consistent pattern of steady improvement across different metrics and languages, although it tends to plateau early in some metrics, indicating good initial learning but limited gains with additional data. In contrast, the PT_Augm_WebNLG model often demonstrates early improvements but struggles to maintain performance with more extensive data, often reaching a plateau or continuously declining.



***(a)** PT_WebNLG_en model*



***(b)** PT_WebNLG_en_ru model*



***(c)** PT_Augm_WebNLG model*

***Figure 13.** BERT F1 Experiments for Welsh*

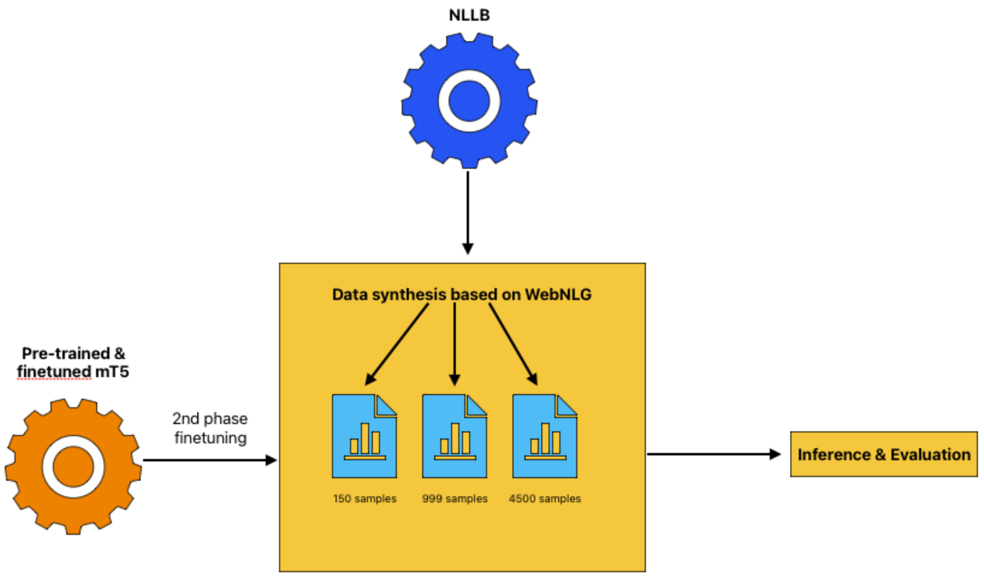


***Figure 14.** Proposed experiments for evaluating our model in zero-shot setting*

# 4 Results

We report our evaluation results in the tables below. Our experiments consist of four models. We denote the further pre-trained models using the filtered OPUS-100 dataset with the prefix *PT* and then the finetuning languages as a postfix. Our experimentation settings for the models can be summarized from the following list

**PT_WebNLG_en** : A model further pre-trained on OPUS-100 and finetuned on the English WebNLG dataset

**PT_WebNLG_en_ru** : A model pre-trained on the OPUS-100 dataset and finetuned with a mixture of Russian and English WebNLG datasets

**PT_Augm_WebNLG** : A finetuned model using synthesized data from the English and Russian WebNLG datasets and augmented the data using the OPUS-100 languages and German.

**Pure_mT5** : A model without pre-training and finetuned based on the best-performing strategy. This model was finetuned according to the PT_Augm_WebNLG recipe.

Furthermore, for the models that we performed in the second finetuning stage, we kept the original name and added the number of samples we used. For simplicity, in the following list, we showcase the non-zero shot generation for which the methodology is explained in section 3.2.10. We denote as **Model** the aforementioned models such as PT_WebNLG_en, PT_WebNLG_en_ru, PT_Augm_WebNLG excluding the Pure_mT5 model.

**Model+150** : A model finetuned on 150 augmented Irish, Maltese, and Welsh samples.

**Model+999** : A model finetuned on 999 augmented Irish, Maltese, and Welsh samples.

**Model+4500** : A model finetuned on 4500 augmented Irish, Maltese, and Welsh samples.

## 4.1 Best performing models

This section provides a detailed comparison of the experiments conducted and their results, focusing on the performance of the models for Breton, Maltese, Irish, and Welsh. The analysis includes the evaluation metrics BLEU, ROUGE, METEOR, TER, chrF++, BERTScore, and PARENT F1.

For Breton, the scores of the models are illustrated in table 7. The best model was the $Pure\_mT5$ since it scored the highest in most metrics. The model's performance was prevalent in the BertScore metrics since it generated semantically closer text than the other models. Another metric that assists the case of $Pure\_mT5$ is METEOR, which considers not only exact matches but also word stems and synonyms for comparison, which often attribute the model's capacity by generating semantically similar text. In generating the most similar text to the reference text, $PT\_Augm\_WebNLG$ did the best job. Metrics such as BLEU, ROUGE, and TER generally capture accuracy in using specific words and phrases and their order and structure. $PT\_WebNLG\_en$ surprisingly score the highest chrF++ score, matching more n-grams than any other model and better preserved character-level information.

In the case of Maltese, the table 5 shows the scores of the models. The best-performing models were the $Pure\_mT5$ and $PT\_Augm\_WebNLG$, as they achieved the highest scores in most key metrics, indicating better quality generations than the reference texts. Specifically, the $Pure\_mT5$ model outperformed all others across BLEU and ROUGE metrics. The $PT\_Augm\_WebNLG$ and models that underwent further pre-training and finetuning also performed well by generating text that matched better at a subword level. These models also demonstrated better performance on TER scores than $Pure\_mT5$, except for $PT\_WebNLG\_en$ and $PT\_WebNLG\_en\_150$. Lastly, for METEOR, most models achieved more or less similar performance. Further pre-training and finetuning helped the models achieve better performance on a granular level in Maltese, while the second stage of finetuning seemed to have helped most models enrich their ability to find synonymous words.

For Irish, the scores of the models are shown in table 4. The best-performing model for Irish was the $PT\_WebNLG\_en\_ru + 4500$, as it scored the highest in most metrics, with $PT\_WebNLG\_en\_ru + 999$ coming in as the second best performer. It can be observed that many models, which were further finetuned with noisy data, could generate more diverse forms of words as they were more enriched. On a detailed level, the enriched models performed better on chrF++ than the $Pure\_mT5$. On the other hand, $Pure\_mT5$ had the highest BERT F1 and BERT P scores, indicating that the model generated more semantically similar text than the others. Generally, the $PT\_WebNLG\_en\_ru + 4500$ and the $PT\_Augm\_WebNLG$ variants showed better performance, as Irish was part of the pre-training languages, and some variants were also included in the further finetuning. These enrichments gave an advantageous position to these variants for the automatic metrics.

For Welsh, the scores of the models are illustrated in table 6. The *PT_Augm_WebNLG* model was the best performer by producing precise and semantically relevant text to the reference text. On chrF++ the *Pure_mT*5 had the lowest score. Since Welsh is also a Celtic family, the model has learned some representation from the pre-training that helped achieve better overlapping of the character-level n-grams and the TER score. Nevertheless, the *Pure_mT*5 also performed the best on ROUGE, BERT F1, and BERT P.

After evaluating models for Breton, Maltese, Irish, and Welsh languages, here are the best-performing models and their strengths:

**For Breton** : *Pure_mT5* was the top model, excelling in five out of nine metrics.

**For Maltese** : *Pure_mT5* and *PT_WebNLG_en_ru + 4500* were the top models, scoring highest in two out of six metrics. Since *Pure_mT5* is under zero-shot, we consider it a superior model.

**For Irish** : *PT_WebNLG_en_ru + 4500* was the best performer, scoring highest in four metrics. But *Pure_mT5* and *PT_Augm_WebNLG* are the best under zero-shot, so we consider them as superior models.

**For Welsh** : *PT_Augm_WebNLG* stood out as the top performer

Overall, the *Pure_mT5* performed the best, and *PT_Augm_WebNLG* was the second best model. Both models performed well in BLEU and ROUGE, while the *PT_WebNLG_en_ru* and its variants performed well in METEOR. We also observed trends in model performance based on their training regimens and datasets used. Additionally, we noted that the automatic metrics provide only one side of the story, and a manual analysis was performed to further support our claims.

| **Models** | **BLEU** | **ROUGE** | **METEOR** | **TER(↓)** | **chrF++** | **BERT_F1** | **BERT_P** | **BERT_R** | **PARENT F1** |
|---|---|---|---|---|---|---|---|---|---|
| *PT_WebNLG_en* | 5.55 | 0.19 | 0.23 | 0.89 | 32.05 | 0.68 | 0.69 | 0.67 | 0.14 |
| *PT_WebNLG_en* + 150 | 5.22 | 0.19 | 0.23 | 0.89 | 32.13 | 0.68 | 0.69 | 0.67 | 0.14 |
| *PT_WebNLG_en* + 999 | 4.62 | 0.2 | 0.24 | 0.9 | 29.86 | 0.64 | 0.66 | 0.63 | **0.15** |
| *PT_WebNLG_en* + 4500 | 7.57 | 0.23 | 0.25 | 0.85 | 31.8 | 0.69 | 0.71 | 0.67 | 0.13 |
| *PT_WebNLG_en_ru* | 5.55 | 0.19 | 0.23 | 0.88 | 32.05 | 0.68 | 0.69 | 0.67 | 0.14 |
| *PT_WebNLG_en_ru* + 150 | 7.17 | 0.22 | **0.26** | 0.85 | 32.83 | 0.69 | 0.71 | **0.68** | 0.13 |
| *PT_WebNLG_en_ru* + 999 | 7.23 | 0.22 | **0.26** | 0.84 | **32.95** | 0.7 | 0.71 | **0.68** | 0.13 |
| *PT_WebNLG_en_ru* + 4500 | 7.58 | **0.23** | **0.26** | **0.82** | 32.93 | 0.7 | 0.72 | **0.68** | 0.13 |
| *PT_Augm_WebNLG* | **7.59** | 0.21 | 0.23 | **0.82** | 29.67 | 0.69 | 0.71 | 0.67 | 0.1 |
| *PT_Augm_WebNLG* + 150 | 6.2 | 0.19 | 0.22 | 0.86 | 29.99 | 0.68 | 0.7 | 0.67 | 0.09 |
| *PT_Augm_WebNLG* + 999 | 6.72 | 0.2 | 0.23 | 0.86 | 30.47 | 0.69 | 0.7 | 0.67 | 0.09 |
| *PT_Augm_WebNLG* + 4500 | 6.7 | 0.2 | 0.23 | 0.86 | 30.43 | 0.69 | 0.7 | 0.67 | 0.09 |
| *Pure_mT*5 | 6.52 | 0.22 | 0.22 | 0.92 | 27.12 | **0.71** | **0.74** | 0.67 | 0.09 |

***Table 4.*** *Evaluation Metrics for Irish (GA)*

| Models | BLEU | ROUGE | METEOR | TER(↓) | chrF++ | PARENT F1 |
|---|---|---|---|---|---|---|
| *PT_WebNLG_en* | 5.72 | 0.2 | 0.24 | 0.92 | 34.75 | **0.15** |
| *PT_WebNLG_en* + 150 | 5.22 | 0.19 | 0.23 | 0.92 | 34.7 | 0.14 |
| *PT_WebNLG_en* + 999 | 7.21 | 0.23 | 0.26 | 0.9 | 34.43 | 0.13 |
| *PT_WebNLG_en* + 4500 | 7.52 | 0.23 | 0.26 | **0.89** | 34.48 | 0.13 |
| *PT_WebNLG_en_ru* | 5.22 | 0.19 | 0.24 | 0.9 | 35.16 | 0.15 |
| *PT_WebNLG_en_ru* + 150 | 6.71 | 0.23 | 0.27 | 0.9 | 35.91 | 0.13 |
| *PT_WebNLG_en_ru* + 999 | 7.01 | 0.23 | 0.27 | 0.91 | 35.92 | 0.14 |
| *PT_WebNLG_en_ru* + 4500 | 7.31 | 0.24 | **0.28** | 0.91 | **36** | 0.14 |
| *PT_Augm_WebNLG* | 8.02 | 0.23 | 0.24 | **0.89** | 31.16 | 0.09 |
| *PT_Augm_WebNLG* + 150 | 6.41 | 0.22 | 0.25 | **0.89** | 34.49 | 0.1 |
| *PT_Augm_WebNLG* + 999 | 6.9 | 0.23 | 0.26 | **0.89** | 34.74 | 0.1 |
| *PT_Augm_WebNLG* + 4500 | 6.91 | 0.23 | 0.26 | 0.91 | 34.82 | 0.1 |
| *Pure_mT5* | **8.68** | **0.25** | 0.26 | 0.91 | 29.27 | 0.1 |

***Table 5.*** *Evaluation Metrics for Maltese (MT)*

| Models | BLEU | ROUGE | METEOR | TER(↓) | chrF++ | BERT_F1 | BERT_P | BERT_R | PARENT_F1 |
|---|---|---|---|---|---|---|---|---|---|
| *PT_WebNLG_en* | 5.97 | 0.2 | 0.23 | 0.93 | 34.26 | 0.71 | 0.72 | 0.7 | **0.13** |
| *PT_WebNLG_en* + 150 | 5.33 | 0.19 | 0.23 | 0.93 | 34.16 | 0.7 | 0.71 | 0.7 | **0.13** |
| *PT_WebNLG_en* + 999 | 4.61 | 0.19 | 0.25 | 0.89 | 31.29 | 0.66 | 0.67 | 0.65 | **0.13** |
| *PT_WebNLG_en* + 4500 | 7.44 | 0.23 | 0.25 | 0.92 | 33.79 | 0.71 | 0.72 | 0.7 | 0.11 |
| *PT_WebNLG_en_ru* | 5.5 | 0.19 | 0.24 | 0.9 | 34.53 | 0.71 | 0.71 | **0.71** | 0.13 |
| *PT_WebNLG_en_ru* + 150 | 7.15 | 0.23 | 0.26 | 0.9 | 35.26 | 0.72 | 0.73 | **0.71** | 0.12 |
| *PT_WebNLG_en_ru* + 999 | 7.23 | 0.23 | **0.27** | 0.86 | 35.3 | 0.72 | 0.73 | **0.71** | 0.12 |
| *PT_WebNLG_en_ru* + 4500 | 7.71 | 0.24 | **0.27** | 0.86 | **35.41** | 0.72 | 0.74 | **0.71** | 0.12 |
| *PT_Augm_WebNLG* | **8.04** | 0.23 | 0.24 | **0.85** | 31.71 | **0.73** | 0.74 | **0.71** | 0.09 |
| *PT_Augm_WebNLG* + 150 | 6.52 | 0.21 | 0.24 | 0.89 | 31.79 | 0.71 | 0.72 | 0.7 | 0.08 |
| *PT_Augm_WebNLG* + 999 | 7.05 | 0.22 | 0.24 | 0.86 | 32.23 | 0.71 | 0.72 | 0.7 | 0.08 |
| *PT_Augm_WebNLG* + 4500 | 6.88 | 0.22 | 0.24 | 0.86 | 32.14 | 0.7 | 0.72 | 0.7 | 0.09 |
| *Pure_mT5* | 7.75 | **0.25** | 0.24 | 0.94 | 29.31 | **0.73** | **0.76** | 0.7 | 0.09 |

***Table 6.*** *Evaluation Metrics for Welsh (CY)*

| Models | BLEU | ROUGE | METEOR | TER(↓) | chrF++ | BERT_F1 | BERT_P | BERT_R | PARENT F1 |
|---|---|---|---|---|---|---|---|---|---|
| *PT_WebNLG_en* | 5.89 | 0.2 | **0.23** | 0.89 | **34.4** | 0.71 | 0.71 | 0.7 | **0.13** |
| *PT_WebNLG_en_ru* | 5.22 | 0.19 | **0.23** | 0.89 | 34.68 | 0.71 | 0.71 | 0.7 | **0.13** |
| *PT_Augm_WebNLG* | **6.8** | **0.23** | 0.22 | **0.87** | 29.99 | 0.73 | 0.75 | 0.71 | 0.08 |
| *Pure_mT5* | 6.16 | **0.23** | **0.23** | 0.91 | 28.54 | **0.74** | **0.77** | **0.72** | 0.07 |

***Table 7.*** *Evaluation Metrics for Breton (BR)*

## 4.2 Manual analysis

This section evaluates our models' generation capabilities by measuring the counts of words in each generated file and creating a distribution for each language found in that file. We perform a manual analysis of each model's generated text files. Using a bag-of-words approach and a language detection tool [12], we want to evaluate whether the highest-performing models generated most of the text in the language in which they were evaluated. This approach does not consider a text's fluency and coherence. The objective is to evaluate

[12]https://fasttext.cc/

distributions of languages present in a text and find insights about the models that generated better under a zero-shot setting.

In the tables below, you can find the language distributions for Welsh (see table 8), Maltese (see table 9), Breton (see table 10), and Irish (see table 11). We provide the language distribution for each evaluation of all variants of our models. These files contain the generated triples in the target language. Often, the models generated in languages other than the target language, a phenomenon known as code-switching. In this evaluation, we are not comparing the model's output with reference texts; our goal is to observe the knowledge the model has acquired through the various training and finetuning recipes.

We will start our approach hierarchically, considering first the top-performing models based on the automatic evaluation metrics for each language.

For Breton, the $Pure\_mT5$ performed the best in the automatic metrics, and in the manual analysis, Breton has 36.84% word coverage from all the generated text, which is the second highest, while $PT\_Augm\_WebNLG$ performed the highest with word coverage in Breton of 42.11%. It validates that the training regimen of finetuning the models with noisy data based on the OPUS-100 languages + German helped them perform better at zero-shot generation. We suspect the reason why the $PT\_Augm\_WebNLG$ scored higher might be attributed to the extra pre-training of the model. On the other hand, the $PT\_WebNLG\_en$ model that was trained only on the English WebNLG dataset has the word coverage concentrated more in English but still contains some words from Arabic and Hebrew, which are attributed to the pre-training regimen. We also observe that from all the other languages, for Breton, the models generated in fewer languages than the other ones, which might also be attributed to the uniqueness of the language since we did not train or finetune with many relative languages.

For Irish, the *PT_WebNLG_en_ru + 4500* model achieved the highest scores in automatic metrics, followed by the *PT_WebNLG_en_ru + 999* model. The *PT_Augm_WebNLG* had the highest word coverage in Irish at 33.33%, with *Pure_mT5* having the second-highest coverage at 27.78%. Most models primarily generated text in Irish or English, occasionally incorporating words from other languages such as Arabic, Hebrew, Amharic, and sometimes French, Italian, Breton, or Russian. Notably, *PT_WebNLG_en* predominantly generated text in English (94.21%), while models like *PT_Augm_WebNLG* and *Pure_mT5* had higher Irish coverage due to their extensive finetuning regimens. Some cross-linguistic influences were observed, such as including Breton and Russian words.

Another interesting case is the *PT_WebNLG_en*+4500 model, which is finetuned in Russian. It occasionally generates a small proportion of words in Russian, a phenomenon not seen in the other models. These findings suggest that comprehensive finetuning and pre-training on diverse language sets significantly enhance the models' ability to generate text in Irish. It is worth noting that no postprocessing step was applied to minimize the code-switching observed in most models.

For Maltese based on automatic metrics the best models were the *Pure_mT5* and *PT_WebNLG_en_ru + 4500*. The *PT_Augm_WebNLG* model achieved the highest Maltese word coverage at 38.89%, and models *Pure_mT5*, *PT_Augm_WebNLG + 4500*, *PT_Augm_WebNLG + 999*, and *PT_Augm_WebNLG + 150* shared the second-highest coverage at 31.58%. Most models primarily generated text in English and Maltese, with occasional words in other languages. This demonstrates that a robust finetuning regimen significantly enhances Maltese text generation. Alternatively, an instruction-tuned model like T5 (50) may have been better for generating text in the target language.

The top-performing models for Welsh are *PT_Augm_WebNLG* and *PT_WebNLG_en_ru + 4500* based on automatic metrics. The *Pure_mT5* model achieved the highest Welsh word coverage at 31.58%. Further finetuning significantly enhances Welsh word generation, although some language mixing occurs.

We can generally assume that high-performing models in automatic metrics achieve greater coverage in the target language being evaluated. This suggests that models capable of generating more fluent text with better n-gram overlap in the target language also tend to produce a larger proportion of text in that language. The augmented WebNLG dataset (28) provided useful insights for the model to differentiate between different languages better than other models. Finally, high performance in automatic metrics does not always guarantee high word coverage in the evaluated language, especially for models trained mainly on English data.

| **Model Name** | **English** | **Welsh** | **Portuguese** | **Arabic** | **Hebrew** | **Amharic** | **Breton** | **Russian** |
|---|---|---|---|---|---|---|---|---|
| *PT_WebNLG_en* in Welsh | 89.00% | 1.62% | 5.15% | 1.42% | 1.65% | | 2.06% | |
| *PT_WebNLG_en + 150 in Welsh* | 81.64% | 10.20% | 2.07% | 1.01% | 2.02% | | 3.06% | |
| *PT_WebNLG_en + 999 in Welsh* | 78.74% | 15.44% | | 3.06% | | 0.55% | 2.21% | |
| *PT_WebNLG_en + 4500 in Welsh* | 75.00% | 10.95% | | 3.48% | 2.35% | 3.22% | 5.00% | |
| *PT_WebNLG_en_ru in Welsh* | 78.29% | 16.67% | | 1.49% | 1.49% | 1.49% | | 0.57% |
| *PT_WebNLG_en_ru + 150 in Welsh* | 73.06% | 16.67% | | 1.30% | 1.30% | 1.30% | 5.56% | 0.81% |
| *PT_WebNLG_en_ru + 999 in Welsh* | 78.91% | 15.44% | | 1.37% | 1.37% | | 2.21% | 0.7% |
| *PT_WebNLG_en_ru + 4500 in Welsh* | 71.22% | 21.00% | | 1.23% | 1.23% | 0.1% | 5.00% | 0.22% |
| *PT_Augm_WebNLG in Welsh* | 71.23% | 21.05% | | 1.23% | 1.23% | | 5.26% | |
| *PT_Augm_WebNLG + 150 in Welsh* | 70.73% | 26.32% | | 1.23% | 1.22% | 0.5% | | |
| *PT_Augm_WebNLG + 999 in Welsh* | 71.02% | 21.05% | | 1.23% | 1.23% | 0.21% | 5.26% | |
| *PT_Augm_WebNLG + 4500 in Welsh* | 70.56% | 26.32% | | 1.23% | 1.23% | 0.66% | | |
| *Pure_mT5 in Welsh* | 64.79% | 31.58% | | 2.28% | 1.35% | | | |

***Table 8.*** *Language Distribution by Model in Welsh*

| **Model Name** | **English** | **Maltese** | **Arabic** | **Hebrew** | **Amharic** | **French** | **Italian** | **Breton** | **Russian** |
|---|---|---|---|---|---|---|---|---|---|
| *PT_WebNLG_en in Maltese* | 90.27% | 1.58% | 1.44% | 1.45% | | 5.26% | | | |
| *PT_WebNLG_en + 150 in Maltese* | 92.11% | 5.05% | 0.81% | 2.03% | | | | | |
| *PT_WebNLG_en + 999 in Maltese* | 85.00% | 10.53% | 1.53% | 1.53% | 1.41% | | | | |
| *PT_WebNLG_en + 4500 in Maltese* | 62.68% | 25.00% | 1.68% | 1.68% | 1.68% | | 5.00% | 5.00% | |
| *PT_WebNLG_en_ru in Maltese* | 70.16% | 27.78% | 0.80% | 0.81% | | | | | 0.45% |
| *PT_WebNLG_en_ru + 150 in Maltese* | 69.12% | 27.78% | 1.20% | 1.20% | 0.15% | | | | 0.55% |
| *PT_WebNLG_en_ru + 999 in Maltese* | 74.57% | 22.22% | 1.30% | 1.30% | | | | | 1.21% |
| *PT_WebNLG_en_ru + 4500 in Maltese* | 69.43% | 22.27% | 1.20% | 1.23% | | | 5.00% | | 0.87% |
| *PT_Augm_WebNLG in Maltese* | 57.74% | 38.89% | 1.02% | 1.02% | 1.33% | | | | |
| *PT_Augm_WebNLG + 150 in Maltese* | 65.49% | 31.58% | 1.14% | 1.14% | 0.65% | | | | |
| *PT_Augm_WebNLG + 999 in Maltese* | 65.81% | 31.58% | 1.14% | 1.14% | 0.33% | | | | |
| *PT_Augm_WebNLG + 4500 in Maltese* | 65.08% | 31.58% | 1.14% | 1.14% | 0.34% | | | | |
| *Pure_mT5 in Maltese* | 63.13% | 31.58% | 1.05% | 1.05% | 1.05% | 2.14% | | | |

***Table 9.*** *Language Distribution by Model in Maltese*

| **Model Name** | **English** | **Portuguese** | **Arabic** | **Hebrew** | **French** | **Breton** | **Russian** |
|---|---|---|---|---|---|---|---|
| *PT_WebNLG_en* in Breton | 94.74% | 1.75% | 1.06% | 2.45% | | | |
| *PT_WebNLG_en_ru in Breton* | 84.89% | | 3.33% | 0.90% | | 10.53% | 0.35% |
| *PT_Augm_WebNLG in Breton* | 50.87% | | 0.88% | 0.88% | 5.26% | 42.11% | |
| *Pure_mT5 in Breton* | 55.98% | | | 1.92% | 5.26% | 36.84% | |

***Table 10.*** *Language Distribution by Model in Breton*

| **Model Name** | **English** | **Irish** | **Arabic** | **Hebrew** | **Amharic** | **French** | **Italian** | **Breton** | **Russian** |
|---|---|---|---|---|---|---|---|---|---|
| *PT_WebNLG_en* in Irish | 94.21% | 1.96% | 1.32% | 2.51% | | | | | |
| *PT_WebNLG_en + 150 in Irish* | 79.16% | 16.67% | 2.23% | 0.11% | 2.23% | | | | |
| *PT_WebNLG_en + 999 in Irish* | 76.57% | 22.22% | 1.53% | 1.53% | 1.21% | | | | |
| *PT_WebNLG_en + 4500 in Irish* | 71.86% | 14.95% | 0.73% | 1.23% | 1.23% | | 5.00% | 5.00% | |
| *PT_WebNLG_en_ru in Irish* | 77.23% | 22.22% | 1.30% | 1.30% | 1.30% | | | | 0.55% |
| *PT_WebNLG_en_ru + 150 in Irish* | 77.00% | 22.22% | 1.39% | 1.39% | 0.45% | | | | 0.33% |
| *PT_WebNLG_en_ru + 999 in Irish* | 76.1% | 22.22% | 1.39% | 1.39% | 0.05% | | | | 0.42% |
| *PT_WebNLG_en_ru + 4500 in Irish* | 75.37% | 16.00% | 1.33% | 1.33% | | 4.00% | | | 0.63% |
| *PT_Augm_WebNLG in Irish* | 65.66% | 33.33% | 1.11% | 1.11% | 1.11% | | | | |
| *PT_Augm_WebNLG + 150 in Irish* | 77.02% | 22.22% | 1.30% | 1.30% | 0.76% | | | | |
| *PT_Augm_WebNLG + 999 in Irish* | 72.07% | 27.78% | 1.20% | 1.20% | 0.15% | | | | |
| *PT_Augm_WebNLG + 4500 in Irish* | 76.87% | 22.22% | 1.30% | 1.30% | 0.91% | | | | |
| *Pure_mT5 in Irish* | 71.17% | 27.78% | 1.20% | 1.20% | 1.05% | | | | |

***Table 11.*** *Language Distribution by Model in Irish*

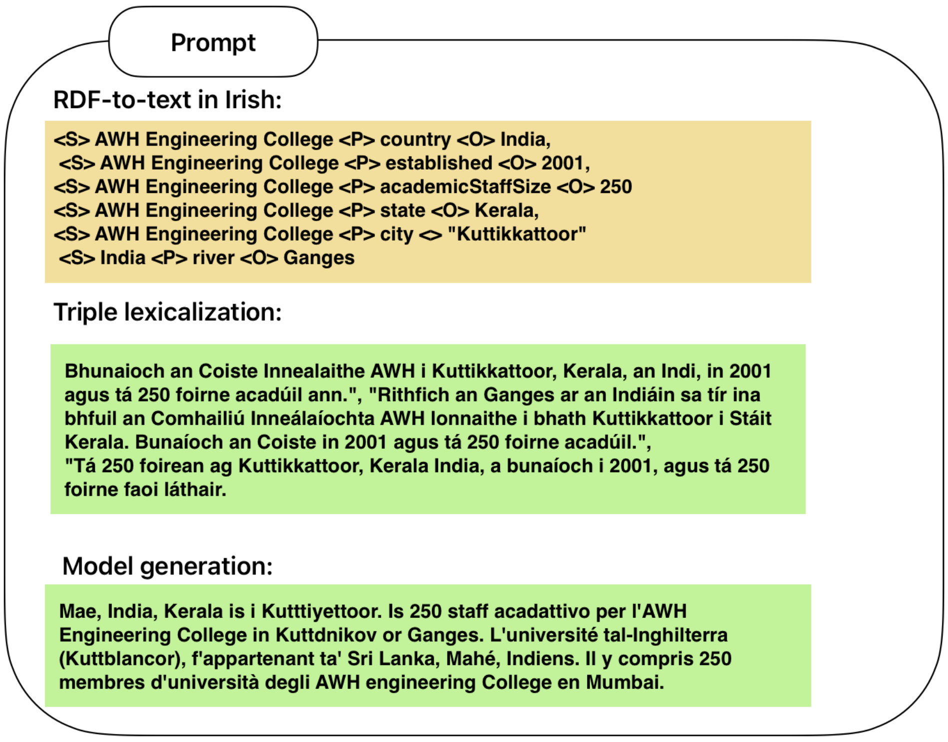
## 4.3 Text generation observations

In this section, we present some examples that we found interesting when manually inspecting the generated text and deemed as noteworthy examples for the scope of this research. In the first observation, we present the triples, lexicalization, and model generation for that specific triple in Breton in figure 18. The $PT\_WebNLG\_en$ model cannot generate text in other languages, leading to code-switching and mostly generating in English.

In the $PT\_WebNLG\_en + 4500$, we observe in figure 15 that the model generates in a mixture of English, Italian, Maltese, and French, with hints of Arabic and Hebrew.

In figure 16, we can see a generation example with Maltese. The model performs similarly in terms of confusion, but we also noticed it hallucinates with certain facts, such as *"a footballer f'pres Cullin"*.

In the generation process, we identified that the model makes certain typos, such as Antonis SamarAS, when the correct ones would be Antonis Samaras and Konstantinos Mitsiotakidis when the correct version is Konstantinos Mitsotakis. This synthetic dataset contains the information correctly. Looking at generations across the models, they accidentally generate Arabic, English, Hebrew, and Amharic, which can be attributed to the pretraining for models with less finetuning examples and data like $PT\_WebNLG\_en$ and $PT\_WebNLG\_en\_ru$, and to the finetuning regimen for models with more finetuning data and diverse examples such as $PT\_Augm\_WebNLG$ and $Pure\_mT5$. More examples of generation can be found in the appendix(see section F)



Prompt

RDF-to-text in Irish:

<S> AWH Engineering College <P> country <O> India,
<S> AWH Engineering College <P> established <O> 2001,
<S> AWH Engineering College <P> academicStaffSize <O> 250
<S> AWH Engineering College <P> state <O> Kerala,
<S> AWH Engineering College <P> city <> "Kuttikkattoor"
<S> India <P> river <O> Ganges

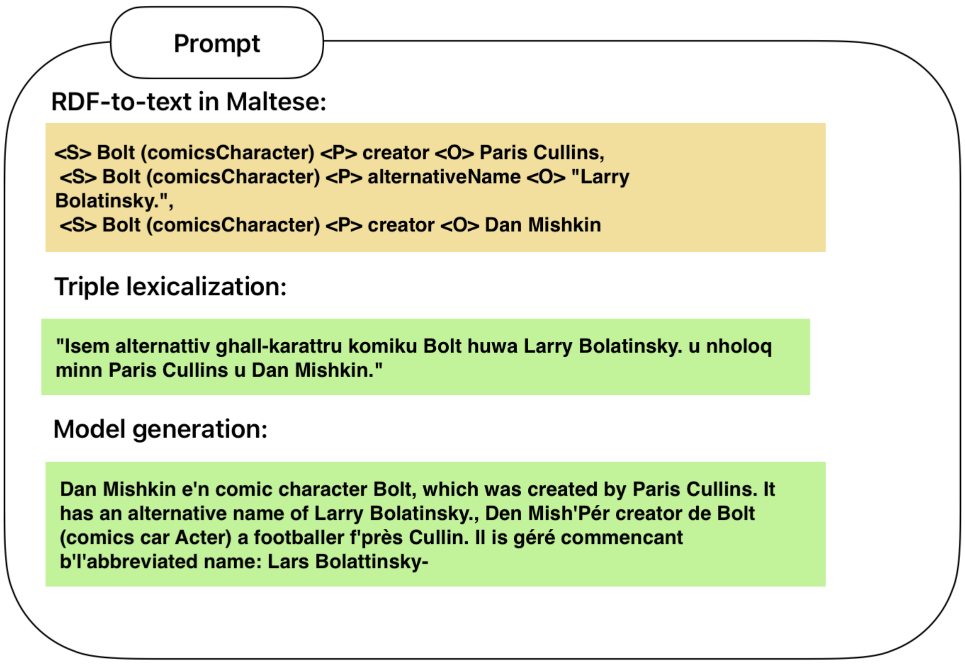
Triple lexicalization:

Bhunaioch an Coiste Innealaithe AWH i Kuttikkattoor, Kerala, an Indi, in 2001 agus tá 250 foirne acadúil ann.", "Rithfich an Ganges ar an Indiáin sa tír ina bhfuil an Comhailiú Inneálaíochta AWH lonnaithe i bhath Kuttikkattoor i Stáit Kerala. Bunaíoch an Coiste in 2001 agus tá 250 foirne acadúil.",
"Tá 250 foirean ag Kuttikkattoor, Kerala India, a bunaíoch i 2001, agus tá 250 foirne faoi láthair.

Model generation:

Mae, India, Kerala is i Kutttiyettoor. Is 250 staff acadattivo per l'AWH Engineering College in Kuttdnikov or Ganges. L'université tal-Inghilterra (Kuttblancor), f'appartenant ta' Sri Lanka, Mahé, Indiens. Il y compris 250 membres d'università degli AWH engineering College en Mumbai.

***Figure 15.*** *Example generating in Irish. Triple lexicalization refers to the reference text*



Prompt

RDF-to-text in Maltese:

<S> Bolt (comicsCharacter) <P> creator <O> Paris Cullins,
<S> Bolt (comicsCharacter) <P> alternativeName <O> "Larry Bolatinsky.",
<S> Bolt (comicsCharacter) <P> creator <O> Dan Mishkin

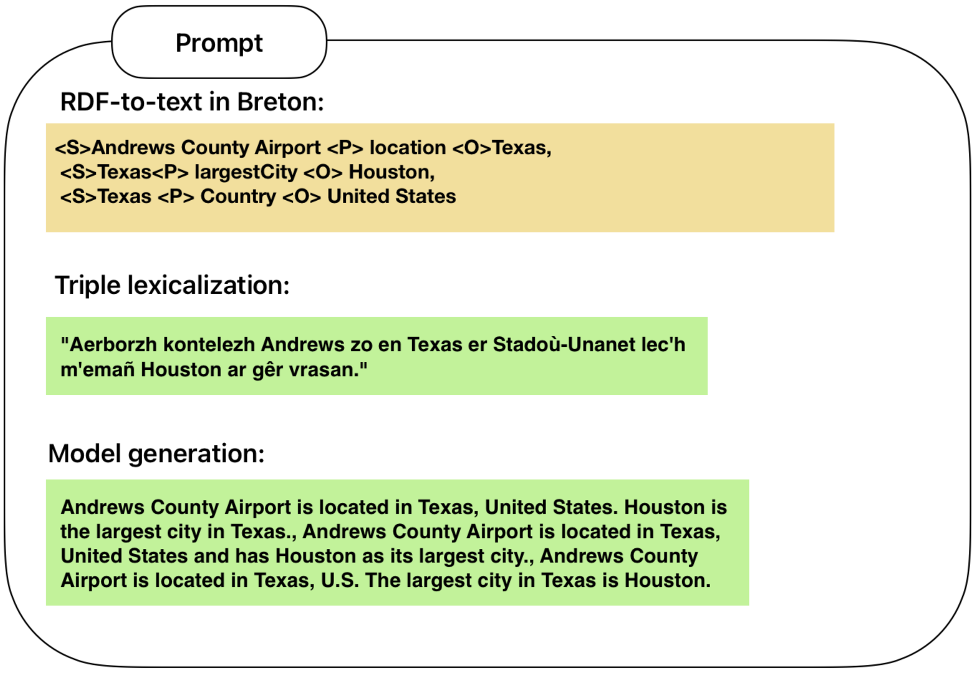
Triple lexicalization:

"Isem alternattiv ghall-karattru komiku Bolt huwa Larry Bolatinsky. u nholoq minn Paris Cullins u Dan Mishkin."

Model generation:

Dan Mishkin e'n comic character Bolt, which was created by Paris Cullins. It has an alternative name of Larry Bolatinsky., Den Mish'Pér creator de Bolt (comics car Acter) a footballer f'près Cullin. Il is géré commencant b'l'abbreviated name: Lars Bolattinsky-

***Figure 16.*** *Example generating in Maltese. Triple lexicalization refers to the reference text*


Prompt

RDF-to-text in Breton:

<S>Andrews County Airport <P> location <O>Texas,
<S>Texas<P> largestCity <O> Houston,
<S>Texas <P> Country <O> United States

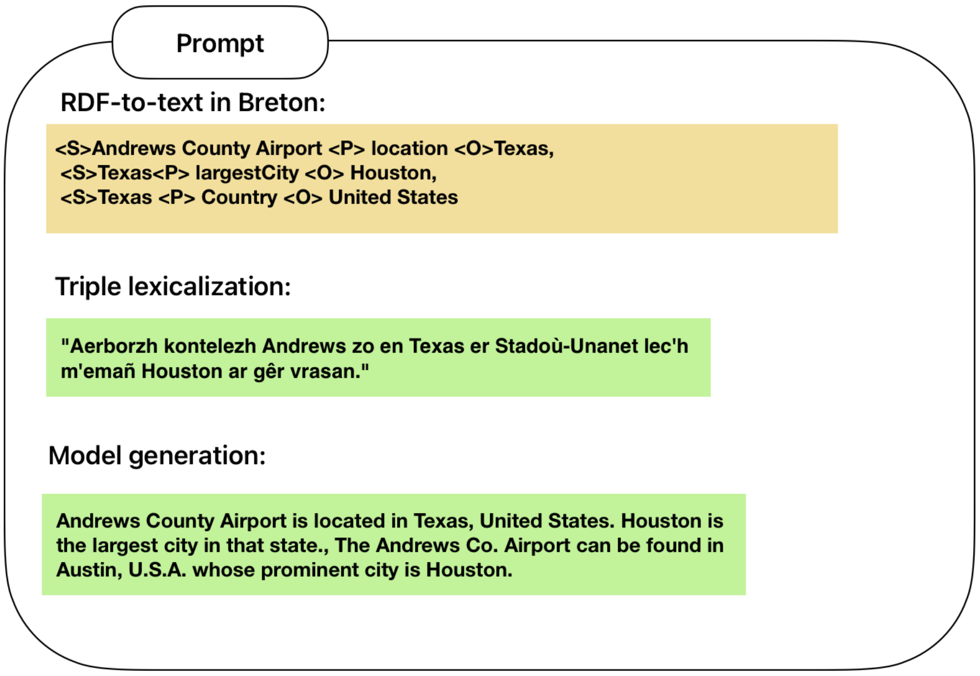
Triple lexicalization:

"Aerborzh kontelezh Andrews zo en Texas er Stadoù-Unanet lec'h m'emañ Houston ar gêr vrasan."

Model generation:

Andrews County Airport is located in Texas, United States. Houston is the largest city in Texas., Andrews County Airport is located in Texas, United States and has Houston as its largest city., Andrews County Airport is located in Texas, U.S. The largest city in Texas is Houston.

***Figure 17.*** *Example of generating in Breton with the old decoding strategy. Triple lexicalization refers to the reference text*



<S>Andrews County Airport <P> location <O>Texas,
<S>Texas<P> largestCity <O> Houston,
<S>Texas <P> Country <O> United States

"Aerborzh kontelezh Andrews zo en Texas er Stadoù-Unanet lec'h m'emañ Houston ar gêr vrasan."

Andrews County Airport is located in Texas, United States. Houston is the largest city in that state., The Andrews Co. Airport can be found in Austin, U.S.A. whose prominent city is Houston.

***Figure 18.*** *Example of generating in Breton with the new decoding strategy. Triple lexicalization refers to the reference text*

## 4.4 Comparison with other WebNLG 2023 models

As seen in the following tables, our best-performing model, $Pure\_mT5$, highlighted in grey, was compared against other competing models in the WebNLG 2023 competition. The comparisons focus on BLEU, chrF++, and TER metrics for Maltese and additional BERT-based metrics for the other languages. Despite our model's significant improvements over the baseline, as discussed in section 4.1, it still performs less than models that underwent extensive training in the WebNLG target languages. This performance gap can be attributed to our model operating in a zero-shot setting, lacking targeted finetuning in the specific languages the evaluation requires. Our model's tendency to generate text in English, even when instructed otherwise, further impacted its performance.

| Models | BLEU | chrF++ | TER (↓) |
|---|---|---|---|
| DCU-NLG-PBN | 21.27 | 0.52 | 0.65 |
| IREL | 16.49 | 0.47 | 0.7 |
| CUNI-Wue | 14.02 | 0.45 | 0.78 |
| Amazon+Zero | 15.60 | 0.42 | 0.67 |
| Pure_mT5 | 8.68 | 0.29 | 0.91 |

***Table 12.*** *Evaluation Metrics for Maltese Generation*

| Models | BLEU | chrF++ | TER (↓) | BERT_P | BERT_R | BERT_F1 |
|---|---|---|---|---|---|---|
| Amazon+Zero | 9.92 | 0.33 | 0.76 | 0.77 | 0.73 | 0.75 |
| CUNI-Wue | 10.09 | 0.33 | 0.8 | 0.76 | 0.73 | 0.74 |
| Pure_mT5 | 6.16 | 0.28 | 0.91 | 0.77 | 0.72 | 0.74 |

***Table 13.*** *Evaluation Metrics for Breton Generation*

| Models | BLEU | chrF++ | TER (↓) | BERT_P | BERT_R | BERT_F1 |
|---|---|---|---|---|---|---|
| DCU-NLG-PBN | 25.11 | 0.55 | 0.64 | 0.83 | 0.83 | 0.83 |
| IREL | 20.97 | 0.49 | 0.67 | 0.82 | 0.8 | 0.81 |
| CUNI-Wue | 17.00 | 0.45 | 0.79 | 0.79 | 0.78 | 0.79 |
| Amazon+Zero | 10.7 | 0.36 | 0.77 | 0.78 | 0.75 | 0.76 |
| Pure_mT5 | 7.75 | 0.29 | 0.94 | 0.76 | 0.7 | 0.73 |

***Table 14.*** *Evaluation Metrics for Welsh Generation*

| **Models** | **BLEU** | **chrF++** | **TER (↓)** | **BERT_P** | **BERT_R** | **BERT_F1** |
|---|---|---|---|---|---|---|
| DCU-NLG-PBN | 20.4 | 0.51 | 0.69 | 0.81 | 0.8 | 0.81 |
| DCU/TCD-FORGe | 16.66 | 0.44 | 0.75 | 0.79 | 0.76 | 0.77 |
| IREL | 15.66 | 0.44 | 0.73 | 0.8 | 0.77 | 0.78 |
| CUNI-Wue | 15.87 | 0.43 | 0.78 | 0.78 | 0.77 | 0.77 |
| Amazon+Zero | 11.63 | 0.36 | 0.74 | 0.78 | 0.74 | 0.76 |
| Pure_mT5 | 6.52 | 0.27 | 0.92 | 0.74 | 0.67 | 0.71 |

***Table 15.*** *Evaluation Metrics for Irish Generation*

Our best-performing model $Pure\_mT5$, despite being a small variant and evaluated in a zero-shot setting, demonstrates its potential by achieving the same score with the $Amazon + Zero$ model on BERT P score, the same BERT F1 with the $CUNI\_Wue$ model in Breton. For Irish and Welsh, we also observe that the *Pure_mT5*, even though it performs worse in BertScore, the performance is in close range to the other models. Lastly, for Maltese, the performance of our model is dramatically worse. The results indicate that with more extensive training and finetuning in the target languages, the model's performance could significantly improve, potentially closing the gap with the leading models in the WebNLG 2023 competition. Moreover, training a larger variant of the $mT5$ could have significant improvements, but we were limited by hardware capacity. The comparisons highlight areas for future work, which we will discuss in more detail in section 6.

## 5 Conclusion & Discussion

This paper discusses an experimental approach for improving data-to-text generation for under-resourced languages under zero-shot and non-zero-shot using the WebNLG benchmark dataset (28). Our experiments were designed to measure the impact of pre-training and finetuning choices in addressing the challenges of generating coherent text in languages with limited resources.

We utilized two primary datasets: the WebNLG corpus and the OPUS-100 dataset. The WebNLG corpus contains structured data in RDF triples paired with textual descriptions in multiple languages, making it ideal for training models on data-to-text tasks. The OPUS-100 is a parallel corpus encompassing 100 languages, providing a richer multilingual training regimen crucial for enriching the models' knowledge of the under-resourced languages. The selection of languages focused on the same family as those represented in the WebNLG challenge, without including the target languages to enable generation in a zero-shot setting.

Data preprocessing was a critical step in the experimental setup for both datasets. For the OPUS-100, this involved cleaning the dataset removing low-quality translations and duplicated text to ensure better quality and consistency. In some instances, language pair reversal was also required, as we aimed to enrich the decoder of our model responsible for text generation. The WebNLG dataset also transformed, including handling the structured nature of the triples to ensure data uniformity and consistency, reshuffling to prevent the model from relying on the order of the triples and converting camelCase to multi-word expressions with average spacing to align the data more closely with natural language. Distinct delimiters such as (<**S**>, <**O**>, and <**P**>) are used to separate subjects, objects, and predicates to improve the model's task comprehension. Finally, RDF triples were serialized by concatenating the subject, predicate, and object into a single string.

The core model used for the experiment was a pre-trained state-of-the-art transformer-based model with an encoder and decoder architecture, namely mT5. We tested various training regimens for mT5 using different datasets and finetuning approaches to evaluate their impact on text generation accuracy.

For our experimental setup, we tested many different training regimens for mT5. The models were trained with the OPUS-100 dataset in all experiments except the last experiment. Initially, we finetuned the pre-trained model using only the English WebNLG dataset (*PT_WebNLG_en*). Then, we trained the model in Russian, which resulted in catastrophic forgetting (115). For this reason, we mixed the Russian and English WebNLG datasets and finetuned from the beginning the mT5 (*PT_WebNLG_en_ru*). Next, we augmented the lexicalizations of the WebNLG dataset to create a WebNLG dataset based on the filtered OPUS-100 languages and German (*PT_Augm_WebNLG*). To further research the impact of the finetuning, we performed a second stage finetuning process on the aforementioned models using augmented data from the WebNLG dataset. The models were tested with different samples to explore how the metric evolution of the models enhanced the ability to perform zero-shot text generation. Lastly, based on the best-performing model, we finetuned an mT5 model with the best settings from the beginning without pre-training it with the OPUS-100 dataset (*Pure_mT*5). This step was done to evaluate the impact of the pre-training.

Our results highlighted that for Breton, the *PT_Augm_WebNLG* model achieved the highest BLEU and TER scores, indicating superior text generation accuracy and fewer required edits, while the *Pure_mT*5 model had the highest chrF++ score. Both models, however, performed poorly on the PARENT metric, reflecting less accuracy in generating contextually appropriate text from structured data. For Maltese, the *Pure_mT5* model excelled with the highest BLEU and ROUGE scores, showcasing its effectiveness in generating accurate and fluent text. Across all languages, models incorporating mixed-language finetuning and data augmentation, such as the *PT_WebNLG_en_ru* variants, demonstrated improvement in METEOR scores. The results suggest that while finetuning with diverse datasets enhances certain metrics, challenges remain in achieving high contextual accuracy and coherence in underrepresented languages.

Additionally, with manual analysis and text generation examples, in sections 4.2 and 4.3, we showed that all models suffered code-switching but data augmentation improved performance in the automatic metrics. This happens partly because of mT5 under zero-shot generation (89) and because we believe the model did not learn to follow the prompts provided during the finetuning. Generally, the model finetuned in many languages did not learn as much as we would expect to generate text in unseen languages. This is because the model might have learned language invariant representations that do not assist it during the generation process (116). Next, we will answer the research questions we introduced at the beginning of this paper and provide a comprehensive answer on a factual basis.

- Main research question:

  Q1: How can language families affect the NLG process?

- Following subquestions:

  SQ1: Is further pre-training necessary, or does it have diminishing returns?

  SQ2: How does finetuning with noisy data impact the performance of mT5 in the text generation tasks?

  SQ3: How does our solution compare to other participating WebNLG models?

We will first focus on answering the subquestions to answer our main research question. For the SQ1, we realized that after extensive experiments with many different scenarios and using the same model, we conclude that language families are not so beneficial in pre-trained models like the mT5, while always considering a good finetuning regimen as discussed in section 4. For example, the best-performing models followed the same finetuning regimen. However, one was pre-trained using the OPUS-100, and the other was not. If we exclude the models trained in the WebNLG target languages, we have the *Pure_mT5* came on top in Maltese, Breton. The

*PT_Augm_WebNLG* came first in Welsh, and together with *Pure_mT*5, they came first for the Irish generation. However, we consider the *Pure_mT*5 superior because OPUS-100 contained Irish. With a few words, for an LLM like mT5, the further pre-training seemed to hurt performance and added no real benefit. We believe a pre-trained instruction-tuned model like T5 (50) would have seen higher gains from the pre-training. The reason is that it is not a multilingual model pre-trained on a massive multilingual mC4 corpus (89) so it would only learn the relevant languages, but it is also instruction tuned, meaning the model can be navigated to perform a task better than mT5 in zero-shot setting.

For SQ2, we focus on finetuning with a noisy data regimen. We observe that the *PT_WebNLG_en* with the simplest finetuning strategy achieved the poorest results because it did not learn to generate in the target language under zero-shot even though it was trained with the high-quality data provided by WebNLG (28). On the other hand, the *PT_Augm_WebNLG* and *Pure_mT*5, which were finetuned with augmented noisy data, saw significantly better performance in zero-shot generation. The models that did not perform zero-shot generation and were trained with the augmented data for Irish, Breton, and Maltese still experienced good performance compared to *PT_WebNLG_en*. Overall, in our experiments, we conclude that the noisy data assisted the model to perform generally better.

For SQ3, our model performed worst in section 4.4 in the WebNLG 2023 challenge. Last but not least, we conclude for our main research question that in our use case where we used the smallest variant of the LLM model mT5, which is not finetuned on any downstream task, language families when finetuning may have played an important role in the zero-shot setting for the text overlapping metrics but of course there was the limitation of instruction following that limited our model to understand the task at hand. We believe that data augmentation with the language of families helped to direct the model in generating better text in unseen tasks, but of course, there would have been greater improvements with a bigger model and dataset as well as an instruction-tuned model.

# 6 Future work

Even though in this project there is an attempt for human evaluation by presenting some text generation observations in section 4.3 and performing qualitative analysis in section 4.2, the results of incorporating human evaluation for the final project assessment is something that would benefit this paper. For example, a metric like the BLEU is widely adopted and easy to use for diagnostic evaluations. However, it is essential to acknowledge the limitations highlighted by recent structured reviews, such as the work by Reiter (117), which shows that BLEU is primarily effective for MT system-level evaluations and less so for NLG or text-level assessments. For this reason, it's really important to include human assessments to ensure a more comprehensive evaluation of our system's performance.

Unlike automated metrics, human evaluation offers an understanding of context, coherence, fluency, and error identification, which are crucial for evaluating natural language generation (NLG) systems. The planned human evaluation will use the quality criteria established in the WebNLG challenge, specifically focusing on fluency, absence of omissions, absence of additions, and unnecessary repetition, with a scoring system from 1 to 5. Unlike human evaluation performed by a group of native speakers, the qualitative analysis done in this paper was based on one person using ready-made solutions to analyze the text and then reading the files for inconsistencies, which is not ideal.

Another important area of improvement is to train more variants of the mT5 model and incorporate more extensive finetuning in target languages and families of languages. The limitations due to hardware capacity suggest that more powerful hardware could significantly improve model performance, potentially bridging the gap with leading models in the WebNLG 2023 competition. Also, by having more resources, we could transfer our parameter settings to the largest variant of mT5 since they would potentially fit the data better and perform better at zero-shot generation.

Moreover, in this project, we created an augmented version of the OPUS-100 dataset by balancing the data of undersampled languages in the distribution by increasing the dataset size 3-fold. In the future, it would be interesting to experiment with this dataset for pre-training and observe its impact. Although the current experiments show diminishing returns in certain cases from additional pre-training, it is essential to investigate the optimal balance and methods for incorporating new data. This could involve developing more sophisticated data preprocessing techniques and experimenting with different model configurations to maximize the added data's benefits.

Lastly, we may also explore different finetuning techniques like instruction tuning. Instruction tuning involves the finetuning datasets of (INSTRUCTION, OUTPUT) pairs (66). It can enhance the model's capability to follow complex instructions and generate more accurate outputs. This finetuning approach could be ideal even in our current scenario since we are convinced that, plenty of times, the model could not follow its instructions to generate in zero-shot. We consider this to be one of the most serious limitations of our model when generating under a zero-shot setting. Instruction tuning is often overlooked and not mentioned as a general issue by the other competing models in the WebNLG challenge (28; 118).

## References

[1] P. Joshi, S. Santy, A. Budhiraja, K. Bali, and M. Choudhury, "The state and fate of linguistic diversity and inclusion in the NLP world," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 6282–6293. [Online]. Available: https://aclanthology.org/2020.acl-main.560

[2] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "Creating training corpora for NLG micro-planners," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 179–188. [Online]. Available: https://www.aclweb.org/anthology/P17-1017.pdf

[3] A. Gatt and E. Krahmer, "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation," 2018.

[4] E. Reiter and R. Dale, *Building Natural Language Generation Systems*, ser. Building Natural Language Generation Systems. Cambridge University Press, 2000. [Online]. Available: https://books.google.nl/books?id=qnWQU9C8bDkC

[5] O. Dušek and F. Jurčíček, "Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, K. Erk and N. A. Smith, Eds. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 45–51. [Online]. Available: https://aclanthology.org/P16-2008

[6] E. Reiter and R. Dale, "Building applied natural language generation systems," *Natural Language Engineering*, vol. 3, 03 2002.

[7] J. Yu, E. Reiter, J. Hunter, and C. Mellish, "Choosing the content of textual summaries of large time-series data sets," *Natural*

*Language Engineering*, vol. 13, pp. 25 – 49, 2006. [Online]. Available: https://api.semanticscholar.org/CorpusID:14924557

[8] A. C. Scott, J. E. Clayton, and E. L. Gibson, *A Practical Guide to Knowledge Acquisition*, 1st ed. USA: Addison-Wesley Longman Publishing Co., Inc., 1991.

[9] A. Hartley and C. Paris, "Two sources of control over the generation of software instructions," in *34th Annual Meeting of the Association for Computational Linguistics*. Santa Cruz, California, USA: Association for Computational Linguistics, Jun. 1996, pp. 192–199. [Online]. Available: https://aclanthology.org/P96-1026

[10] H. Dalianis, "Aggregation in natural language generation," *Computational Intelligence*, vol. 15, no. 4, pp. 384–414, 1999.

[11] E. Reiter and R. Dale, "Building applied natural language generation systems," *Natural Language Engineering*, vol. 3, no. 1, pp. 57–87, 1997.

[12] S. W. McRoy, S. Channarukul, and S. S. Ali, "An augmented template-based approach to text realization," *Natural Language Engineering*, vol. 9, pp. 381 – 420, 2003. [Online]. Available: https://api.semanticscholar.org/CorpusID:16074781

[13] J. Bateman, "Enabling technology for multilingual natural language generation: the kpml development environment," *Natural Language Engineering*, vol. 3, 03 1997.

[14] M. Halliday and C. Matthiessen, *An Introduction to Functional Grammar*, 01 2004.

[15] A. Cahill, M. Forst, and C. Rohrer, "Stochastic realisation ranking for a free word order language," in *Proceedings of the Eleventh European Workshop on Natural Language Generation (ENLG 07)*, S. Busemann, Ed. Saarbrücken, Germany: DFKI GmbH, Jun. 2007, pp. 17–24. [Online]. Available: https://aclanthology.org/W07-2303

[16] J. Carroll and S. Oepen, "High efficiency realization for a wide-coverage unification grammar," in *Second International Joint Conference on Natural Language Processing: Full Papers*, 2005. [Online]. Available: https://aclanthology.org/I05-1015

[17] J. Novikova, O. Dušek, and V. Rieser, "The E2E dataset: New challenges for end-to-end generation," in *Proceedings of the 18th*

*Annual SIGdial Meeting on Discourse and Dialogue*, K. Jokinen, M. Stede, D. DeVault, and A. Louis, Eds. Saarbrücken, Germany: Association for Computational Linguistics, Aug. 2017, pp. 201–206. [Online]. Available: https://aclanthology.org/W17-5525

[18] L. Nan, D. Radev, R. Zhang, A. Rau, A. Sivaprasad, C. Hsieh, X. Tang, A. Vyas, N. Verma, P. Krishna, Y. Liu, N. Irwanto, J. Pan, F. Rahman, A. Zaidi, M. Mutuma, Y. Tarabar, A. Gupta, T. Yu, Y. C. Tan, X. V. Lin, C. Xiong, R. Socher, and N. F. Rajani, "Dart: Open-domain structured data record to text generation," 2021.

[19] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, P. Isabelle, E. Charniak, and D. Lin, Eds. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002, pp. 311–318. [Online]. Available: https://aclanthology.org/P02-1040

[20] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, J. Goldstein, A. Lavie, C.-Y. Lin, and C. Voss, Eds. Ann Arbor, Michigan: Association for Computational Linguistics, Jun. 2005, pp. 65–72. [Online]. Available: https://aclanthology.org/W05-0909

[21] T. Sellam, D. Das, and A. Parikh, "BLEURT: Learning robust metrics for text generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, Eds. Online: Association for Computational Linguistics, Jul. 2020, pp. 7881–7892. [Online]. Available: https://aclanthology.org/2020.acl-main.704

[22] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A study of translation edit rate with targeted human annotation," in *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*. Cambridge, Massachusetts, USA: Association for Machine Translation in the Americas, Aug. 8-12 2006, pp. 223–231. [Online]. Available: https://aclanthology.org/2006.amta-papers.25

[23] W. Zhao, M. Peyrard, F. Liu, Y. Gao, C. M. Meyer, and S. Eger,

"Moverscore: Text generation evaluating with contextualized embeddings and earth mover distance," 2019.

[24] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," 2020.

[25] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," 2019.

[26] S. Kweon, Y. Kwon, S. Cho, Y. Jo, and E. Choi, "Open-wikitable: Dataset for open domain question answering with complex reasoning over table," 2023.

[27] V. Zhong, C. Xiong, and R. Socher, "Seq2sql: Generating structured queries from natural language using reinforcement learning," 2017.

[28] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "The WebNLG challenge: Generating text from RDF data," in *Proceedings of the 10th International Conference on Natural Language Generation*, J. M. Alonso, A. Bugarín, and E. Reiter, Eds. Santiago de Compostela, Spain: Association for Computational Linguistics, Sep. 2017, pp. 124–133. [Online]. Available: https://aclanthology.org/W17-3518

[29] O. Dušek, D. M. Howcroft, and V. Rieser, "Semantic noise matters for neural natural language generation," in *Proceedings of the 12th International Conference on Natural Language Generation*, K. van Deemter, C. Lin, and H. Takamura, Eds. Tokyo, Japan: Association for Computational Linguistics, Oct.–Nov. 2019, pp. 421–426. [Online]. Available: https://aclanthology.org/W19-8652

[30] R. Lebret, D. Grangier, and M. Auli, "Neural text generation from structured data with application to the biography domain," 2016.

[31] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 11 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[32] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014.

[Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf

[33] T. Ferreira, I. Calixto, S. Wubben, and E. Krahmer, "Linguistic realisation as machine translation: Comparing different mt models for amr-to-text generation," 01 2017, pp. 1–10.

[34] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2016.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023.

[36] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:49313245

[37] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[38] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[39] E. v. M. a. K. Thiago Castro Ferreira, Chris van der Lee, "Neural data-to-text generation: A comparison between pipeline and end-to-end architectures," 2019. [Online]. Available: https://arxiv.org/abs/1908.09022

[40] R. Puduppully, L. Dong, and M. Lapata, "Data-to-text generation with content selection and planning," *arXiv: Computation and Language*, 2018. [Online]. Available: https://arxiv.org/abs/1809.00582

[41] S. Wiseman, S. Shieber, and A. Rush, "Challenges in data-to-document generation," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, M. Palmer, R. Hwa, and S. Riedel, Eds. Copenhagen, Denmark: Association for Computational Linguistics, sep 2017, pp. 2253–2263. [Online]. Available: https://aclanthology.org/D17-1239

[42] O. Vinyals, M. Fortunato, and N. Jaitly, “Pointer networks,” 2017.

[43] A. Rohrbach, L. A. Hendricks, K. Burns, T. Darrell, and K. Saenko, “Object hallucination in image captioning,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, Eds. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 4035–4045. [Online]. Available: https://aclanthology.org/D18-1437

[44] M. Shanahan, “Talking about large language models,” 2023.

[45] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel, “Palm: Scaling language modeling with pathways,” 2022.

[46] B. Workshop, “Bloom: A 176b-parameter open-access multilingual language model,” 2022, in-depth studies by Le Scao et al. (2022) and Wang et al. (2022a). [Online]. Available: https://arxiv.org/abs/2204.02311

[47] S. Zhang, “Opt: Open pre-trained transformer language models,” 2022. [Online]. Available: https://arxiv.org/abs/2205.01068

[48] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, “Llama: Open and efficient foundation language models,” 2023.

[49] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre, “Training compute-optimal large language models,” 2022.

[50] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2023.

[51] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen, "A survey of large language models," 2023. [Online]. Available: https://arxiv.org/pdf/2303.18223.pdf

[52] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[53] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, M. Walker, H. Ji, and A. Stent, Eds. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 2227–2237. [Online]. Available: https://aclanthology.org/N18-1202

[54] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:160025533

[55] OpenAI, "Gpt-4 technical report," 2023.

[56] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," 2020.

[57] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[58] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," 09 2019.

[59] J. W. Rae, S. Borgeaud, T. Cai, K. Millican, J. Hoffmann, F. Song, J. Aslanides, S. Henderson, R. Ring, S. Young, E. Rutherford, T. Hennigan, J. Menick, A. Cassirer, R. Powell, G. van den Driessche, L. A. Hendricks, M. Rauh, P.-S. Huang, A. Glaese, J. Welbl, S. Dathathri, S. Huang, J. Uesato, J. Mellor, I. Higgins, A. Creswell, N. McAleese,

A. Wu, E. Elsen, S. Jayakumar, E. Buchatskaya, D. Budden, E. Sutherland, K. Simonyan, M. Paganini, L. Sifre, L. Martens, X. L. Li, A. Kuncoro, A. Nematzadeh, E. Gribovskaya, D. Donato, A. Lazaridou, A. Mensch, J.-B. Lespiau, M. Tsimpoukelli, N. Grigorev, D. Fritz, T. Sottiaux, M. Pajarskas, T. Pohlen, Z. Gong, D. Toyama, C. de Masson d'Autume, Y. Li, T. Terzi, V. Mikulik, I. Babuschkin, A. Clark, D. de Las Casas, A. Guy, C. Jones, J. Bradbury, M. Johnson, B. Hechtman, L. Weidinger, I. Gabriel, W. Isaac, E. Lockhart, S. Osindero, L. Rimell, C. Dyer, O. Vinyals, K. Ayoub, J. Stanway, L. Bennett, D. Hassabis, K. Kavukcuoglu, and G. Irving, "Scaling language models: Methods, analysis  insights from training gopher," 2022.

[60] O. Lieber, O. Sharir, B. Lenz, and Y. Shoham, "Jurassic-1: Technical details and evaluation," Aug. 2021.

[61] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. Korthikanti, E. Zhang, R. Child, R. Y. Aminabadi, J. Bernauer, X. Song, M. Shoeybi, Y. He, M. Houston, S. Tiwary, and B. Catanzaro, "Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model," 2022.

[62] R. Thoppilan, D. D. Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, Y. Li, H. Lee, H. S. Zheng, A. Ghafouri, M. Menegali, Y. Huang, M. Krikun, D. Lepikhin, J. Qin, D. Chen, Y. Xu, Z. Chen, A. Roberts, M. Bosma, V. Zhao, Y. Zhou, C.-C. Chang, I. Krivokon, W. Rusch, M. Pickett, P. Srinivasan, L. Man, K. Meier-Hellstern, M. R. Morris, T. Doshi, R. D. Santos, T. Duke, J. Soraker, B. Zevenbergen, V. Prabhakaran, M. Diaz, B. Hutchinson, K. Olson, A. Molina, E. Hoffman-John, J. Lee, L. Aroyo, R. Rajakumar, A. Butryna, M. Lamm, V. Kuzmina, J. Fenton, A. Cohen, R. Bernstein, R. Kurzweil, B. Aguera-Arcas, C. Cui, M. Croak, E. Chi, and Q. Le, "Lamda: Language models for dialog applications," 2022.

[63] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," 2019.

[64] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," 2016.

[65] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto,

O. Vinyals, P. Liang, J. Dean, and W. Fedus, "Emergent abilities of large language models," 2022.

[66] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe, "Training language models to follow instructions with human feedback," 2022.

[67] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023.

[68] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," 2020.

[69] I. McKenzie, A. Lyzhov, A. Parrish, A. Prabhu, A. Mueller, N. Kim, S. Bowman, and E. Perez, "The inverse scaling prize," 2022. [Online]. Available: https://github.com/inverse-scaling/prize

[70] (2023) Common crawl. https://commoncrawl.org/. [Online].

[71] J. Baumgartner, S. Zannettou, B. Keegan, M. Squire, and J. Blackburn, "The pushshift reddit dataset," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 14, no. 1, pp. 830–839, May 2020. [Online]. Available: https://ojs.aaai.org/index.php/ICWSM/article/view/7347

[72] (2023) Wikipedia. https://www.wikipedia.org/. [Online].

[73] E. Nijkamp, B. Pang, H. Hayashi, L. Tu, H. Wang, Y. Zhou, S. Savarese, and C. Xiong, "Codegen: An open large language model for code with multi-turn program synthesis," 2023.

[74] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[75] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," 2022.

[76] V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja, M. Dey, M. S. Bari,

C. Xu, U. Thakker, S. S. Sharma, E. Szczechla, T. Kim, G. Chhablani, N. Nayak, D. Datta, J. Chang, M. T.-J. Jiang, H. Wang, M. Manica, S. Shen, Z. X. Yong, H. Pandey, R. Bawden, T. Wang, T. Neeraj, J. Rozen, A. Sharma, A. Santilli, T. Fevry, J. A. Fries, R. Teehan, T. Bers, S. Biderman, L. Gao, T. Wolf, and A. M. Rush, "Multitask prompted training enables zero-shot task generalization," 2022.

[77] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi, "Self-instruct: Aligning language models with self-generated instructions," 2023.

[78] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu, and G. Wang, "Instruction tuning for large language models: A survey," 2024.

[79] S. Longpre, L. Hou, T. Vu, A. Webson, H. W. Chung, Y. Tay, D. Zhou, Q. V. Le, B. Zoph, J. Wei, and A. Roberts, "The flan collection: Designing data and methods for effective instruction tuning," 2023.

[80] S. Mishra, D. Khashabi, C. Baral, and H. Hajishirzi, "Cross-task generalization via natural language crowdsourcing instructions," 2022.

[81] C. Zhou, P. Liu, P. Xu, S. Iyer, J. Sun, Y. Mao, X. Ma, A. Efrat, P. Yu, L. Yu, S. Zhang, G. Ghosh, M. Lewis, L. Zettlemoyer, and O. Levy, "Lima: Less is more for alignment," 2023.

[82] N. Muennighoff, T. Wang, L. Sutawika, A. Roberts, S. Biderman, T. L. Scao, M. S. Bari, S. Shen, Z.-X. Yong, H. Schoelkopf, X. Tang, D. Radev, A. F. Aji, K. Almubarak, S. Albanie, Z. Alyafeai, A. Webson, E. Raff, and C. Raffel, "Crosslingual generalization through multitask finetuning," 2023.

[83] C. Xu, Q. Sun, K. Zheng, X. Geng, P. Zhao, J. Feng, C. Tao, and D. Jiang, "Wizardlm: Empowering large language models to follow complex instructions," 2023.

[84] P.-N. Kung and N. Peng, "Do models really learn to follow instructions? an empirical study of instruction tuning," 2023.

[85] J. Gao, H. Zhao, C. Yu, and R. Xu, "Exploring the feasibility of chatgpt for event extraction," 2023.

[86] C. Xu, D. Guo, N. Duan, and J. McAuley, "Baize: An open-source chat model with parameter-efficient tuning on self-chat data," 2023.

[87] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," 2023.

[88] Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu, L. Li, and Z. Sui, "A survey on in-context learning," 2023.

[89] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, "mt5: A massively multilingual pre-trained text-to-text transformer," 2021.

[90] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer, "Multilingual denoising pre-training for neural machine translation," 2020.

[91] D. Moussallem, D. Gnaneshwar, T. C. Ferreira, and A.-C. N. Ngomo, "Nabu − multilingual graph-based neural rdf verbalizer," 2020.

[92] "Data-to-text generation for severely under-resourced languages with gpt-3.5: A bit of help needed from google translate," 2023. [Online]. Available: https://aclanthology.org/2023.mmnlg-1.9.pdf

[93] N. Kumar, S. Obaid Ul Islam, and O. Dusek, "Better translation + split and generate for multilingual RDF-to-text (WebNLG 2023)," in *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*, A. Gatt, C. Gardent, L. Cripwell, A. Belz, C. Borg, A. Erdem, and E. Erdem, Eds. Prague, Czech Republic: Association for Computational Linguistics, Sep. 2023, pp. 73–79. [Online]. Available: https://aclanthology.org/2023.mmnlg-1.8

[94] O. Agarwal, M. Kale, H. Ge, S. Shakeri, and R. Al-Rfou, "Machine translation aided bilingual data-to-text generation and semantic parsing," in *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, Eds. Dublin, Ireland (Virtual): Association for Computational Linguistics, 12 2020, pp. 125–130. [Online]. Available: https://aclanthology.org/2020.webnlg-1.13

[95] S. Mille, R. Carlini, A. Burga, and L. Wanner, "FORGe at SemEval-2017 task 9: Deep sentence generation based on a sequence of graph transducers," in *Proceedings of the 11th*

*International Workshop on Semantic Evaluation (SemEval-2017)*, S. Bethard, M. Carpuat, M. Apidianaki, S. M. Mohammad, D. Cer, and D. Jurgens, Eds. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 920–923. [Online]. Available: https://aclanthology.org/S17-2158

[96] B. Trisedya, J. Qi, R. Zhang, and W. Wang, “Gtr-lstm: A triple encoder for sentence generation from rdf data,” 01 2018, pp. 1627–1637.

[97] L. L. Xiaojun Wan, Hongyu Zang. (2017) Pkuwriter. https://synalp.gitlabpages.inria.fr/webnlg-challenge/files/PKUWriter_report.pdf. [Online].

[98] T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, “The 2020 bilingual, bi-directional WebNLG+ shared task: Overview and evaluation results (WebNLG+ 2020),” in *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, Eds. Dublin, Ireland (Virtual): Association for Computational Linguistics, 12 2020, pp. 55–76. [Online]. Available: https://aclanthology.org/2020.webnlg-1.7

[99] Z. Yang, A. Einolghozati, H. Inan, K. Diedrick, A. Fan, P. Donmez, and S. Gupta, “Improving text-to-text pre-trained models for the graph-to-text task,” in *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, Eds. Dublin, Ireland (Virtual): Association for Computational Linguistics, 12 2020, pp. 107–116. [Online]. Available: https://aclanthology.org/2020.webnlg-1.11

[100] X. Li, A. Maskharashvili, S. Jory Stevens-Guille, and M. White, “Leveraging large pretrained models for WebNLG 2020,” in *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, Eds. Dublin, Ireland (Virtual): Association for Computational Linguistics, 12 2020, pp. 117–124. [Online]. Available: https://aclanthology.org/2020.webnlg-1.12

[101] Q. Guo, Z. Jin, N. Dai, X. Qiu, X. Xue, D. Wipf, and Z. Zhang, “ $\sqrt{}^{2}$: A plan-and-pretrain approach for knowledge graph-to-text

generation," in *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, T. Castro Ferreira, C. Gardent, N. Ilinykh, C. van der Lee, S. Mille, D. Moussallem, and A. Shimorina, Eds. Dublin, Ireland (Virtual): Association for Computational Linguistics, 12 2020, pp. 100–106. [Online]. Available: https://aclanthology.org/2020.webnlg-1.10

[102] L. Cripwell, A. Belz, C. Gardent, A. Gatt, C. Borg, M. Borg, J. Judge, M. Lorandi, A. Nikiforovskaya, W. Soto-Martinez, and C. Thomson, "The 2023 WebNLG Shared Task on Low Resource Languages Overview and Evaluation Results (WebNLG 2023)," in *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*, Prague, Czech Republic, Sep. 2023. [Online]. Available: https://hal.science/hal-04356939

[103] M. Kazakov, J. Preobrazhenskaya, I. Bulychev, and A. Shain, "WebNLG-interno: Utilizing FRED-t5 to address the RDF-to-text problem (WebNLG 2023)," in *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*, A. Gatt, C. Gardent, L. Cripwell, A. Belz, C. Borg, A. Erdem, and E. Erdem, Eds. Prague, Czech Republic: Association for Computational Linguistics, Sep. 2023, pp. 67–72. [Online]. Available: https://aclanthology.org/2023.mmnlg-1.7

[104] M. Lorandi and A. Belz, "Data-to-text generation for severely under-resourced languages with GPT-3.5: A bit of help needed from Google Translate (WebNLG 2023)," in *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*, A. Gatt, C. Gardent, L. Cripwell, A. Belz, C. Borg, A. Erdem, and E. Erdem, Eds. Prague, Czech Republic: Association for Computational Linguistics, Sep. 2023, pp. 80–86. [Online]. Available: https://aclanthology.org/2023.mmnlg-1.9

[105] M. Post, "A call for clarity in reporting BLEU scores," in *Proceedings of the Third Conference on Machine Translation: Research Papers*, O. Bojar, R. Chatterjee, C. Federmann, M. Fishel, Y. Graham, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, C. Monz, M. Negri, A. Névéol, M. Neves, M. Post, L. Specia, M. Turchi, and K. Verspoor, Eds. Brussels, Belgium: Association

for Computational Linguistics, Oct. 2018, pp. 186–191. [Online]. Available: https://aclanthology.org/W18-6319

[106] M. Popović, "chrF: character n-gram F-score for automatic MT evaluation," in *Proceedings of the Tenth Workshop on Statistical Machine Translation*, O. Bojar, R. Chatterjee, C. Federmann, B. Haddow, C. Hokamp, M. Huck, V. Logacheva, and P. Pecina, Eds. Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 392–395. [Online]. Available: https://aclanthology.org/W15-3049

[107] ——, "chrF++: words helping character n-grams," in *Proceedings of the Second Conference on Machine Translation*, O. Bojar, C. Buck, R. Chatterjee, C. Federmann, Y. Graham, B. Haddow, M. Huck, A. J. Yepes, P. Koehn, and J. Kreutzer, Eds. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 612–618. [Online]. Available: https://aclanthology.org/W17-4770

[108] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81. [Online]. Available: https://aclanthology.org/W04-1013

[109] B. Dhingra, M. Faruqui, A. Parikh, M.-W. Chang, D. Das, and W. W. Cohen, "Handling divergent reference texts when evaluating table-to-text generation," 2019.

[110] B. Zhang, P. Williams, I. Titov, and R. Sennrich, "Improving massively multilingual neural machine translation and zero-shot translation," 2020.

[111] J. Tiedemann, "Parallel data, tools and interfaces in OPUS," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, N. Calzolari, K. Choukri, T. Declerck, M. U. Doğan, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, Eds. Istanbul, Turkey: European Language Resources Association (ELRA), May 2012, pp. 2214–2218. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2012/pdf/463_Paper.pdf

[112] N. Shazeer and M. Stern, "Adafactor: Adaptive learning rates with sublinear memory cost," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine

Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 4596–4604. [Online]. Available: https://proceedings.mlr.press/v80/shazeer18a.html

[113] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, and R. Socher, "Ctrl: A conditional transformer language model for controllable generation," 2019.

[114] N. Team, M. R. Costa-jussà, J. Cross, O. Çelebi, M. Elbayad, K. Heafield, K. Heffernan, E. Kalbassi, J. Lam, D. Licht, J. Maillard, A. Sun, S. Wang, G. Wenzek, A. Youngblood, B. Akula, L. Barrault, G. M. Gonzalez, P. Hansanti, J. Hoffman, S. Jarrett, K. R. Sadagopan, D. Rowe, S. Spruit, C. Tran, P. Andrews, N. F. Ayan, S. Bhosale, S. Edunov, A. Fan, C. Gao, V. Goswami, F. Guzmán, P. Koehn, A. Mourachko, C. Ropers, S. Saleem, H. Schwenk, and J. Wang, "No language left behind: Scaling human-centered machine translation," 2022.

[115] T. Vu, A. Barua, B. Lester, D. Cer, M. Iyyer, and N. Constant, "Overcoming catastrophic forgetting in zero-shot cross-lingual generation," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Y. Goldberg, Z. Kozareva, and Y. Zhang, Eds. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 9279–9300. [Online]. Available: https://aclanthology.org/2022.emnlp-main.630

[116] T. Li and K. Murray, "Why does zero-shot cross-lingual generation fail? an explanation and a solution," 2023.

[117] E. Reiter, "A structured review of the validity of BLEU," *Computational Linguistics*, vol. 44, no. 3, pp. 393–401, Sep. 2018. [Online]. Available: https://aclanthology.org/J18-3002

[118] L. Cripwell, A. Belz, C. Gardent, A. Gatt, C. Borg, M. Borg, J. Judge, M. Lorandi, A. Nikiforovskaya, and W. Soto Martinez, "The 2023 WebNLG shared task on low resource languages. overview and evaluation results (WebNLG 2023)," in *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*, A. Gatt, C. Gardent, L. Cripwell, A. Belz, C. Borg, A. Erdem, and E. Erdem, Eds. Prague, Czech Republic: Association for Computational Linguistics, Sep. 2023, pp. 55–66. [Online]. Available: https://aclanthology.org/2023.mmnlg-1.6

## A OPUS-100 supported pair languages

- af-en: Afrikaans to English
- am-en: Amharic to English
- an-en: Aragonese to English
- ar-en: Arabic to English
- as-en: Assamese to English
- az-en: Azerbaijani to English
- be-en: Belarusian to English
- bg-en: Bulgarian to English
- bn-en: Bengali to English
- br-en: Breton to English
- bs-en: Bosnian to English
- ca-en: Catalan to English
- cs-en: Czech to English
- cy-en: Welsh to English
- da-en: Danish to English
- de-en: German to English
- dz-en: Dzongkha to English
- el-en: Greek to English
- en-eo: English to Esperanto
- en-es: English to Spanish
- en-et: English to Estonian
- en-eu: English to Basque
- en-fa: English to Persian
- en-fi: English to Finnish
- en-fr: English to French
- en-fy: English to Western Frisian
- en-ga: English to Irish
- en-gd: English to Scottish Gaelic
- en-gl: English to Galician
- en-gu: English to Gujarati
- en-ha: English to Hausa
- en-he: English to Hebrew
- en-hi: English to Hindi
- en-hr: English to Croatian
- en-hu: English to Hungarian
- en-hy: English to Armenian
- en-id: English to Indonesian
- en-ig: English to Igbo

- en-is: English to Icelandic
- en-it: English to Italian
- en-ja: English to Japanese
- en-ka: English to Georgian
- en-kk: English to Kazakh
- en-km: English to Khmer
- en-ko: English to Korean
- en-kn: English to Kannada
- en-ku: English to Kurdish
- en-ky: English to Kyrgyz
- en-li: English to Limburgish
- en-lt: English to Lithuanian
- en-lv: English to Latvian
- en-mg: English to Malagasy
- en-mk: English to Macedonian
- en-ml: English to Malayalam
- en-mn: English to Mongolian
- en-mr: English to Marathi
- en-ms: English to Malay
- en-mt: English to Maltese
- en-my: English to Burmese
- en-nb: English to Norwegian Bokmål
- en-ne: English to Nepali
- en-nl: English to Dutch
- en-nn: English to Norwegian Nynorsk
- en-no: English to Norwegian
- en-oc: English to Occitan
- en-or: English to Odia
- en-pa: English to Punjabi
- en-pl: English to Polish
- en-ps: English to Pashto
- en-pt: English to Portuguese
- en-ro: English to Romanian
- en-ru: English to Russian
- en-rw: English to Kinyarwanda
- en-se: English to Northern Sami
- en-sh: English to Serbo-Croatian
- en-si: English to Sinhala
- en-sk: English to Slovak
- en-sl: English to Slovenian
- en-sq: English to Albanian

- en-sr: English to Serbian
- en-sv: English to Swedish
- en-ta: English to Tamil
- en-te: English to Telugu
- en-tg: English to Tajik
- en-th: English to Thai
- en-tk: English to Turkmen
- en-tr: English to Turkish
- en-tt: English to Tatar
- en-ug: English to Uyghur
- en-uk: English to Ukrainian
- en-ur: English to Urdu
- en-uz: English to Uzbek
- en-vi: English to Vietnamese
- en-wa: English to Walloon
- en-xh: English to Xhosa
- en-yi: English to Yiddish
- en-yo: English to Yoruba
- en-zh: English to Chinese
- en-zu: English to Zulu

## B mT5 pre-trained languages

mT5 is pre-trained on the mC4 corpus, covering 101 languages:
Afrikaans, Albanian, Amharic, Arabic, Armenian, Azerbaijani, Basque, Belarusian, Bengali, Bulgarian, Burmese, Catalan, Cebuano, Chichewa, Chinese, Corsican, Czech, Danish, Dutch, English, Esperanto, Estonian, Filipino, Finnish, French, Galician, Georgian, German, Greek, Gujarati, Haitian Creole, Hausa, Hawaiian, Hebrew, Hindi, Hmong, Hungarian, Icelandic, Igbo, Indonesian, Irish, Italian, Japanese, Javanese, Kannada, Kazakh, Khmer, Korean, Kurdish, Kyrgyz, Lao, Latin, Latvian, Lithuanian, Luxembourgish, Macedonian, Malagasy, Malay, Malayalam, Maltese, Maori, Marathi, Mongolian, Nepali, Norwegian, Pashto, Persian, Polish, Portuguese, Punjabi, Romanian, Russian, Samoan, Scottish Gaelic, Serbian, Shona, Sindhi, Sinhala, Slovak, Slovenian, Somali, Sotho, Spanish, Sundanese, Swahili, Swedish, Tajik, Tamil, Telugu, Thai, Turkish, Ukrainian, Urdu, Uzbek, Vietnamese, Welsh, West Frisian, Xhosa, Yiddish, Yoruba, Zulu.

## C mBART pre-trained languages

mBART is pre-trained on the CC25 checkpoint and monolingual data from XLMR dataset. Supported languages are:
Arabic (ar_AR), Czech (cs_CZ), German (de_DE), English (en_XX), Spanish (es_XX), Estonian (et_EE), Finnish (fi_FI), French (fr_XX), Gujarati (gu_IN), Hindi (hi_IN), Italian (it_IT), Japanese (ja_XX), Kazakh (kk_KZ), Korean (ko_KR), Lithuanian (lt_LT), Latvian (lv_LV), Burmese (my_MM), Nepali (ne_NP), Dutch (nl_XX), Romanian (ro_RO), Russian (ru_RU), Sinhala (si_LK), Turkish (tr_TR), Vietnamese (vi_VN), Chinese (zh_CN), Afrikaans (af_ZA), Azerbaijani (az_AZ), Bengali (bn_IN), Persian (fa_IR), Hebrew (he_IL), Croatian (hr_HR), Indonesian (id_ID), Georgian (ka_GE), Khmer (km_KH), Macedonian (mk_MK), Malayalam (ml_IN), Mongolian (mn_MN), Marathi (mr_IN), Polish (pl_PL), Pashto (ps_AF), Portuguese (pt_XX), Swedish (sv_SE), Swahili (sw_KE), Tamil (ta_IN), Telugu (te_IN), Thai (th_TH), Tagalog (tl_XX), Ukrainian (uk_UA), Urdu (ur_PK), Xhosa (xh_ZA), Galician (gl_ES), Slovene (sl_SI)

## D Learning curves and tables

| **Lang Pair / Metric** | **BLEU** | **ROUGE** | **METEOR** | **TER($\downarrow$)** | **chrF++** | **BERT_F1** | **BERT_P** | **BERT_R** |
|---|---|---|---|---|---|---|---|---|
| en-am | 0.65 | 0.01 | 0.08 | 93.66 | 8.31 | 0.90 | 0.91 | 0.88 |
| en-ar | 5.52 | 0.03 | 0.17 | 87.95 | 21.67 | 0.77 | 0.79 | 0.75 |
| en-fr | 6.32 | 0.35 | 0.29 | 84.01 | 22.59 | 0.75 | 0.78 | 0.73 |
| en-ga | 3.66 | 0.40 | 0.29 | 85.22 | 21.27 | 0.77 | 0.81 | 0.74 |
| en-gd | 4.33 | 0.24 | 0.17 | 86.92 | 19.58 | 0.73 | 0.74 | 0.72 |
| en-it | 10.90 | 0.39 | 0.35 | 80.21 | 28.71 | 0.78 | 0.79 | 0.76 |
| en-he | 18.35 | 0.01 | 0.39 | 73.84 | 35.29 | 0.82 | 0.83 | 0.81 |

***Table 16.*** *Detailed Evaluation Metrics for Test Data Across Language Pairs when pre-training*

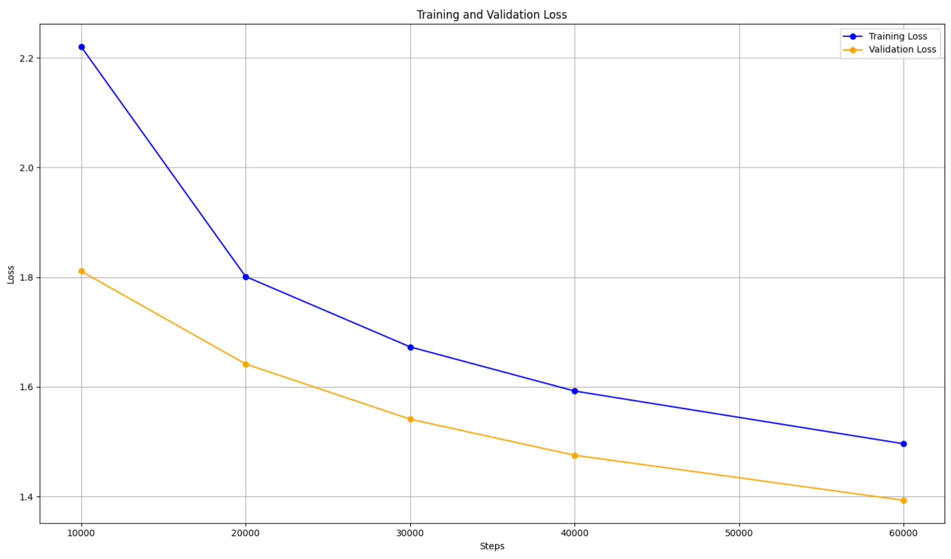


***Figure 19.*** *Training/validation loss when pre-training on the filtered OPUS-100 dataset*

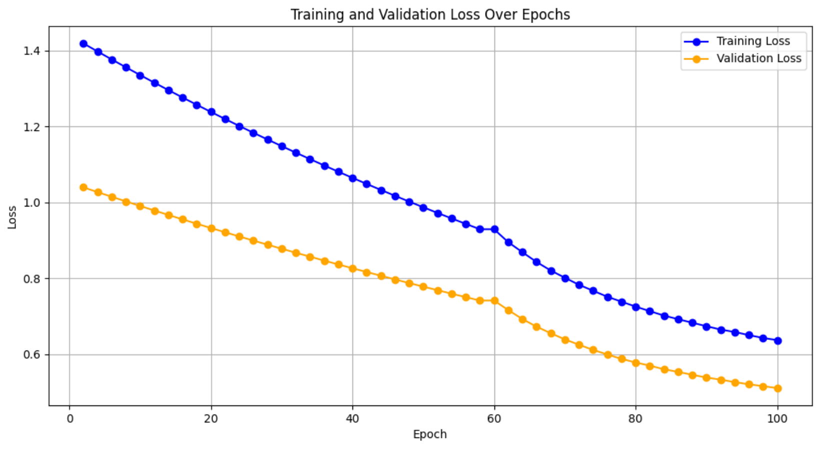


***Figure 20.*** *Training/validation loss when finetuning on English for 100 epochs for the first experiment*

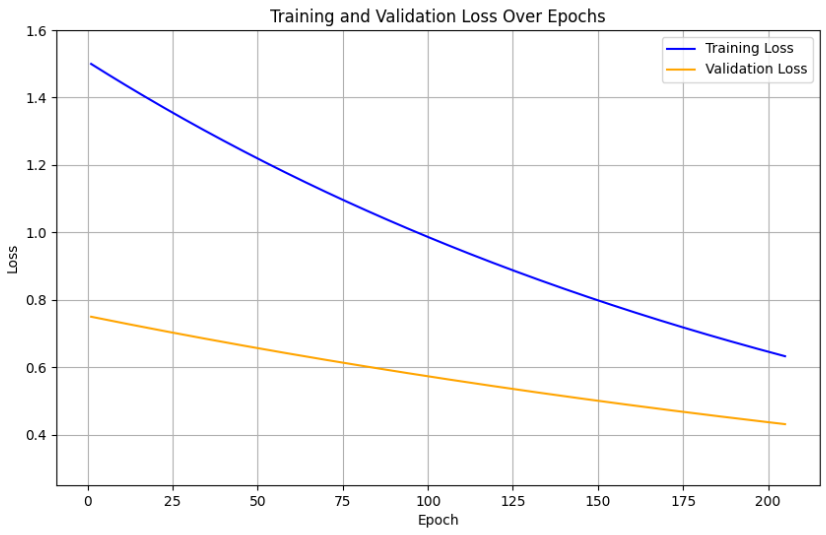


***Figure 21.*** *Training/validation loss when finetuning on English for 200 epochs for the second experiment*

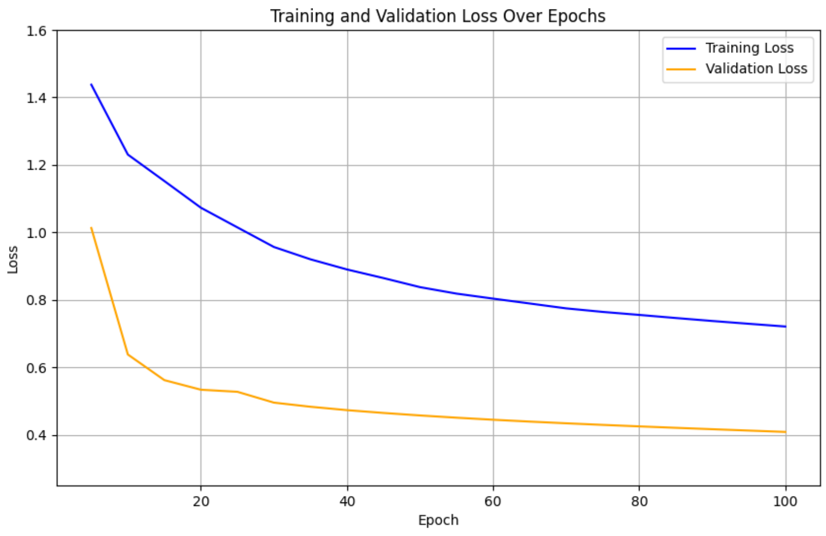
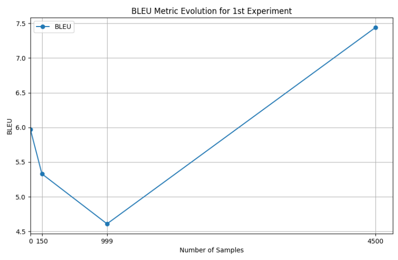
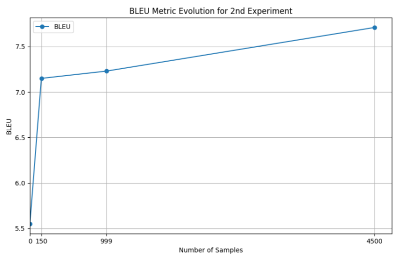


***Figure 22.*** *Training/validation loss when finetuning on OPUS-100+german for 100 epochs for the third experiment*

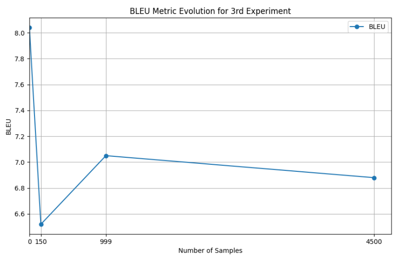
## E Metric evolution of the second stage finetuning



*(a) PT_WebNLG_en model*

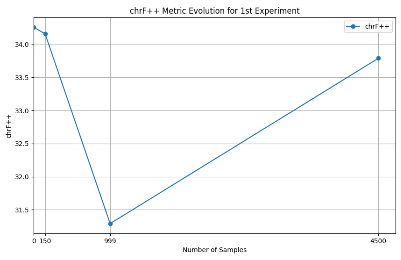


*(b) PT_WebNLG_en_ru model*

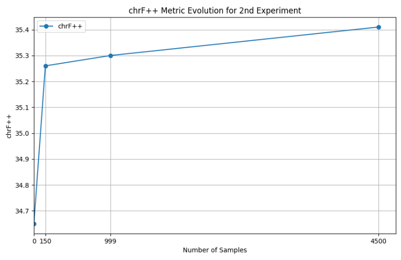


*(c) PT_Augm_WebNLG model*

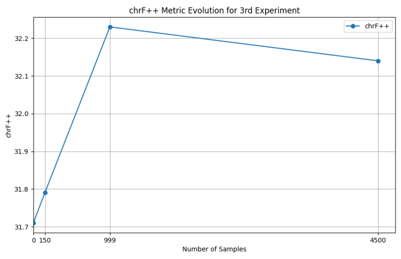
***Figure 23.*** *BLEU Experiments for Welsh*



*(a) PT_WebNLG_en model*

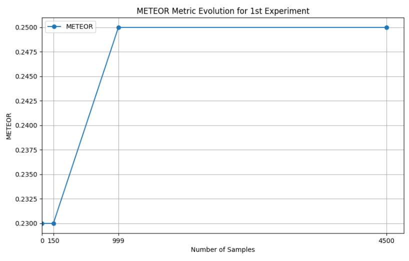


*(b) PT_WebNLG_en_ru model*

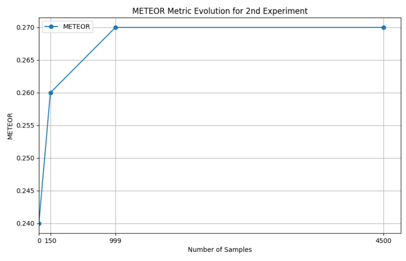


*(c) PT_Augm_WebNLG model*

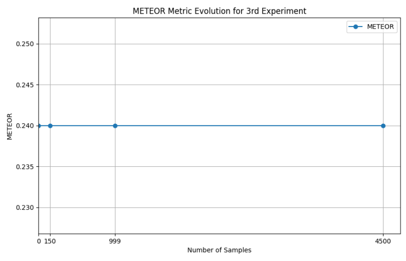
***Figure 24.*** *chrF++ Experiments for Welsh*



***(a)*** *PT_WebNLG_en model*

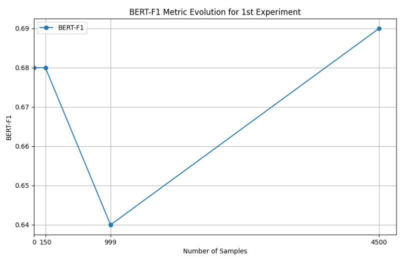


***(b)*** *PT_WebNLG_en_ru model*

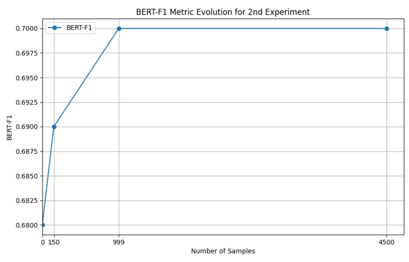


***(c)*** *PT_Augm_WebNLG model*

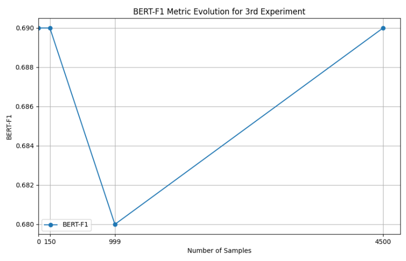
***Figure 25.*** *METEOR Experiments for Welsh*



*(a) PT_WebNLG_en model*

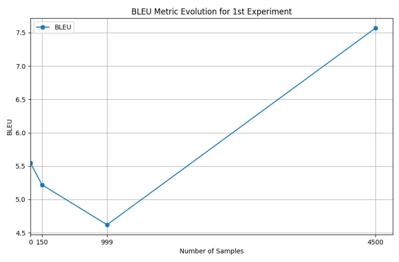


*(b) PT_WebNLG_en_ru model*

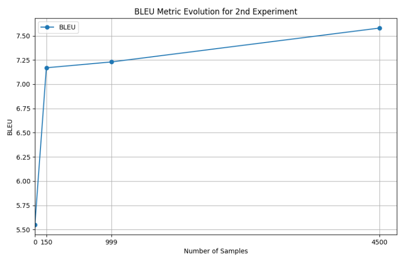


*(c) PT_Augm_WebNLG model*

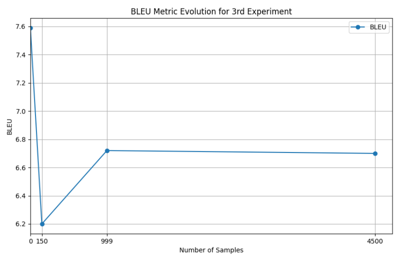
***Figure 26.*** *BERT F1 Experiments for Irish*



*(a) PT_WebNLG_en model*

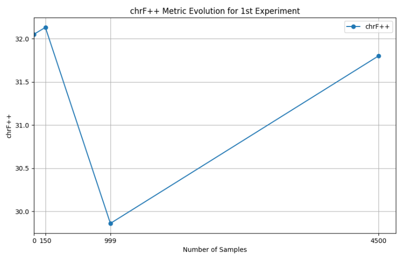


*(b) PT_WebNLG_en_ru model*

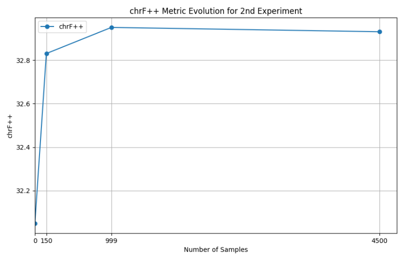


*(c) PT_Augm_WebNLG model*

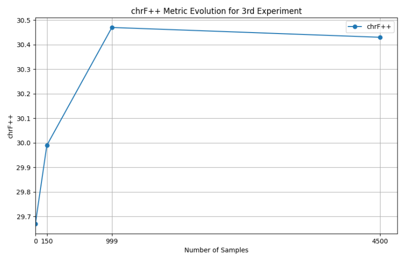
***Figure 27.*** *BLEU Experiments for Irish*



*(a) PT_WebNLG_en model*

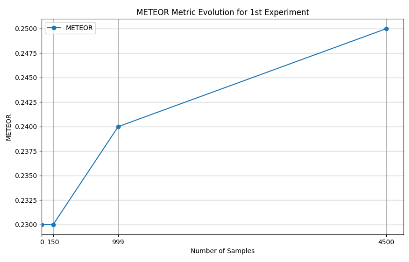


*(b) PT_WebNLG_en_ru model*

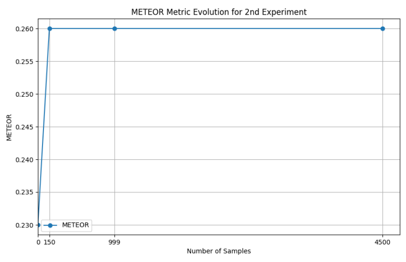


*(c) PT_Augm_WebNLG model*

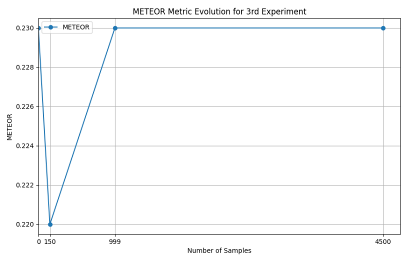
***Figure 28.*** *chrF++ Experiments for Irish*



*(a) PT_WebNLG_en model*

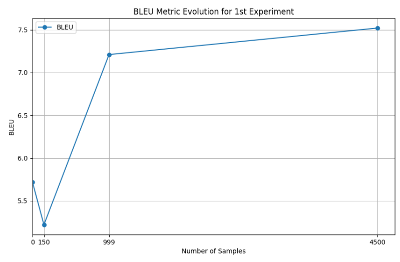


*(b) PT_WebNLG_en_ru model*

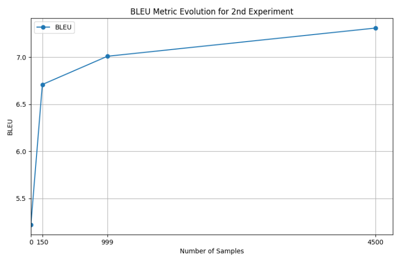


*(c) PT_WebNLG_en_ru model*

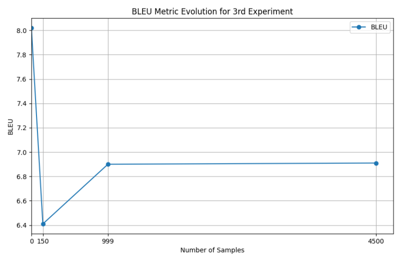
***Figure 29.*** *METEOR Experiments for Irish*



*(a) PT_WebNLG_en model*

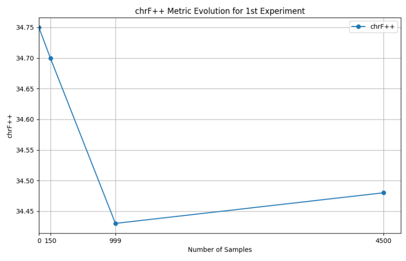


*(b) PT_WebNLG_en_ru model*

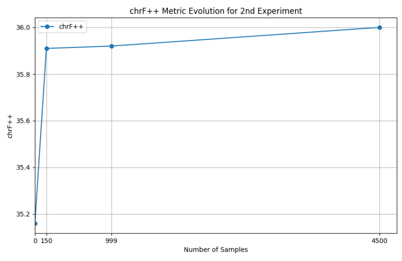


*(c) PT_Augm_WebNLG model*

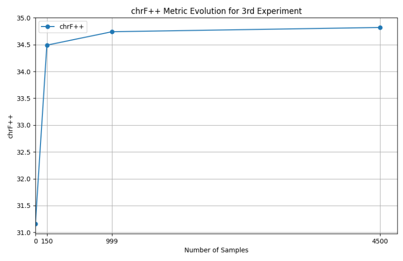
***Figure 30.*** *BLEU Experiments for Maltese*



***(a)*** *PT_WebNLG_en model*

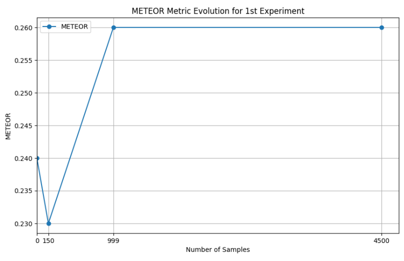


***(b)*** *PT_WebNLG_en_ru model*

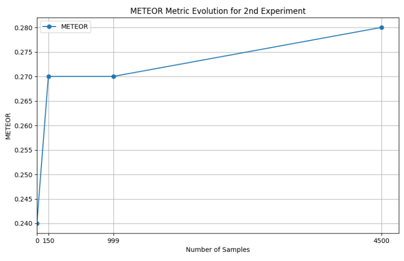


***(c)*** *PT_Augm_WebNLG model*

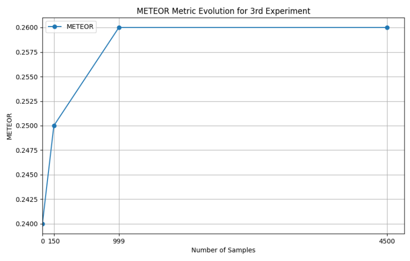
***Figure 31.*** *chrF++ Experiments for Maltese*



*(a) PT_WebNLG_en model*



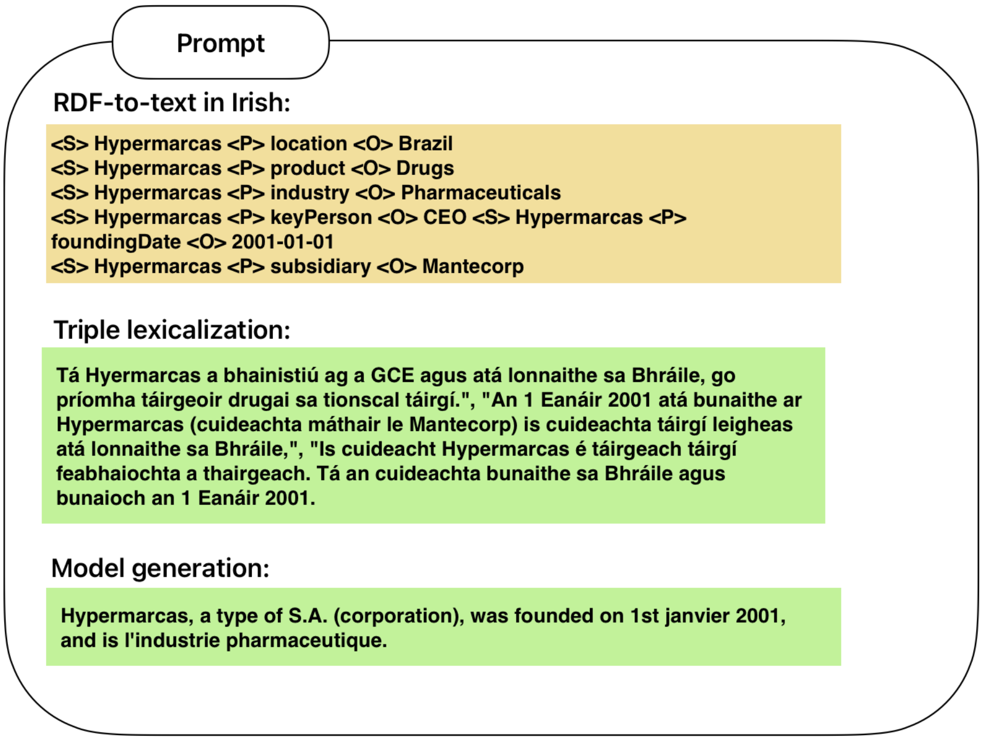
*(b) PT_WebNLG_en_ru model*



*(c) PT_Augm_WebNLG model*

***Figure 32.*** *METEOR Experiments for Maltese*

## F Text generation observation figures



Prompt

RDF-to-text in Irish:

<S> Hypermarcas <P> location <O> Brazil
<S> Hypermarcas <P> product <O> Drugs
<S> Hypermarcas <P> industry <O> Pharmaceuticals
<S> Hypermarcas <P> keyPerson <O> CEO <S> Hypermarcas <P> foundingDate <O> 2001-01-01
<S> Hypermarcas <P> subsidiary <O> Mantecorp

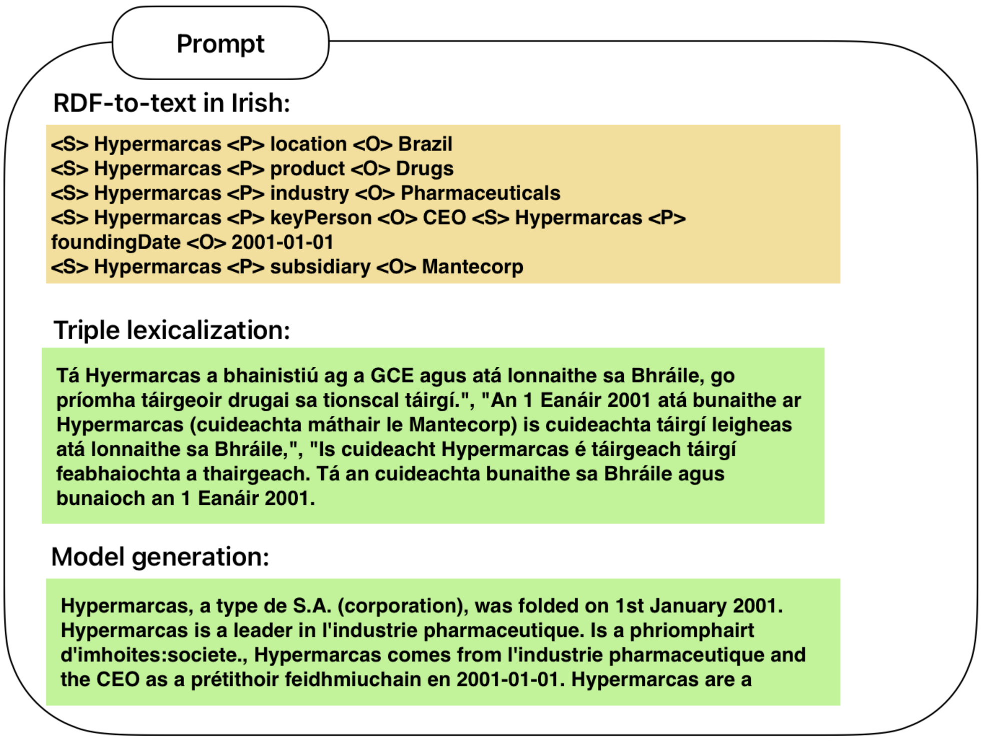
Triple lexicalization:

Tá Hyermarcas a bhainistiú ag a GCE agus atá lonnaithe sa Bhráile, go príomha táirgeoir drugai sa tionscal táirgí.", "An 1 Eanáir 2001 atá bunaithe ar Hypermarcas (cuideachta máthair le Mantecorp) is cuideachta táirgí leigheas atá lonnaithe sa Bhráile,", "Is cuideacht Hypermarcas é táirgeach táirgí feabhaiochta a thairgeach. Tá an cuideachta bunaithe sa Bhráile agus bunaioch an 1 Eanáir 2001.

Model generation:

Hypermarcas, a type of S.A. (corporation), was founded on 1st janvier 2001, and is l'industrie pharmaceutique.

***Figure 33.*** *Example generating in Irish. Triple lexicalization refers to the reference text*



Prompt

RDF-to-text in Irish:

<S> Hypermarcas <P> location <O> Brazil
<S> Hypermarcas <P> product <O> Drugs
<S> Hypermarcas <P> industry <O> Pharmaceuticals
<S> Hypermarcas <P> keyPerson <O> CEO <S> Hypermarcas <P> foundingDate <O> 2001-01-01
<S> Hypermarcas <P> subsidiary <O> Mantecorp

Triple lexicalization:

Tá Hyermarcas a bhainistiú ag a GCE agus atá lonnaithe sa Bhráile, go príomha táirgeoir drugai sa tionscal táirgí.", "An 1 Eanáir 2001 atá bunaithe ar Hypermarcas (cuideachta máthair le Mantecorp) is cuideachta táirgí leigheas atá lonnaithe sa Bhráile,", "Is cuideacht Hypermarcas é táirgeach táirgí feabhaiochta a thairgeach. Tá an cuideachta bunaithe sa Bhráile agus bunaioch an 1 Eanáir 2001.

Model generation:

Hypermarcas, a type de S.A. (corporation), was folded on 1st January 2001. Hypermarcas is a leader in l'industrie pharmaceutique. Is a phriomphairt d'imhoites:societe., Hypermarcas comes from l'industrie pharmaceutique and the CEO as a prétithoir feidhmiuchain en 2001-01-01. Hypermarcas are a

***Figure 34.*** *Example generating in Irish. Triple lexicalization refers to the reference text*