MSc Artificial Intelligence
Master Thesis

# Language control prefixes: Conditional prefix-tuning for efficient multilingual data-to-text generation in low-resource languages

by
David Vos
11295481

July 1, 2022

48 EC
November 2021 - June 2022

*Supervisor:*
Dr. Ing. S. Schelter
*Examiner:*
Dr Ing. S. Schelter
*Second reader:*
Prof. Dr. P.T. Groth

Universiteit van Amsterdam

**Abstract**

Robot journalism is an upcoming field that allows journalists to automate the process of writing simple reports of sports matches, finance or weather. Neural language models have the potential of being more diverse and less expensive than current template-based approaches. Fine-tuning large language models is memory-intensive and difficult for low-resource languages. In this work we show that prefix-tuning can efficiently be used for adapting a language model in a multilingual data-to-text setting, specifically in low-resource scenarios. We then propose language control prefixes as a more effective way of efficiently learning a prefix by enabling learning from language-agnostic features.

# Contents

# Chapter 1

# Introduction

*"If you want creative workers, give them enough time to play."* - John Cleese

## 1.1 Robot journalism

Over the past years, more and more news publishers have employed software-based 'robot journalists'. Human editors usually prefer spending time on unique stories that require human interaction or research. Robot journalists on the other hand, are able to write massive amounts of simple stories in a short period of time. By creating pipelines to automatically write simple reports, humans are enabled to spend more time doing creative work. *The Washington Post* introduced a system that is able to write simple pieces on sports and politics based on structured data. The commercial developments of the relatively simple technology behind robot journalism also paved the way for smaller publishers. For example, the Dutch *Dagblad van het Noorden* uses software to report on all amateur soccer games in their region.

This thesis is written for *DPG Media*, the largest media publisher in the Netherlands and one of the largest in Belgium. DPG Media publishes traditional newspapers like *De Volkskrant*, *AD* and *Trouw*. The company also owns news platforms like *Nu.nl* and video streaming sites such as *VTM Go*. Throughout DPG Media, people are looking into employing software to automate parts of the news publication process. This is done through academic research as well as employing existing solutions for products owned by the company. This thesis is an example of the former, introducing novel contributions to the field of robot journalism while staying relevant to the vision of DPG Media.

## 1.2 Templating and language modelling

The current state-of-the-art robot journalism systems are based on templating. Templates consist of sets of sentences with gaps that are filled based on structured data. By selecting what sentences are relevant based on certain data, a story is generated. This approach is an extremely simple but reliable way of news generation [31]. However, it is labour intensive to create templates for different domains and languages. Additionally, resulting stories will always be limited in diversity, again relying on human input to come up with new sentences or structures.

Employing language models (LMs) is one of the main proposed alternatives to templating. By automatically learning from existing data, the problems of diversity and expensive labour are potentially solved. Neural language models have recently gained widespread attention because of their promising results. Models with a Transformer architecture like GPT-3 by OpenAI [4]

and T5 by Google [28] have shown excellent performance at tasks like summarization, text classification or question-answering. These models are able to solve complex language-related problems without any sort of fine-tuning or adapting the model to the task at hand. However, they have shown limited capabilities for the data-to-text task [18], which is necessary to write news based on structured data.

Data-to-text generation requires a data structure as input and then generates a story based on its content. For news generation, it is very important that the resulting generations are truthful. The output should contain as much information as possible that is present in the input data, while at the same time not generating anything that is not mentioned. Large neural language models struggle with both of these problems, especially when used without adapting the model to the task at hand. Fine-tuning a large pre-trained LM like GPT-3 or T5 is necessary to optimize the model as much as possible for the data-to-text task.

## 1.3 Efficient fine-tuning of pre-trained language models

When fine-tuning an LM for a task in a specific language, all parameters are updated and stored. With the increasing accessibility of deploying large LMs, the amount of storage and computational power necessary can quickly become extremely high. Fine-tuning multiple models for slightly different tasks, domains or languages can have large effects on costs and the environment [11].

Researchers have recently introduced more efficient ways of fine-tuning large pre-trained LMs. Three examples are adapter tuning [25], masking [36] and prefix-tuning [18]. From these methods, prefix-tuning has shown the most promising results in terms of both performance and efficiency.

Prefix-tuning essentially learns a continuous prefix that is prependend to the original data input. The technique uses only 0.1% of the parameters required to fine-tune an entire LM while having similar performance [18]. Upon training time, only the prefix is updated and all LM parameters are kept frozen.

## 1.4 Multilingual prefix-tuning

Following the release of large pre-trained LMs, researchers have extended the concept to a multilingual setting. This means that LMs are not only pre-trained on a single, but on multiple languages. For example mT5 [35] is an instance of T5 [28] but trained on a dataset with over 100 languages. These developments allow for applications and research in multiple languages, especially ones with limited available resources [22].

Multilingual models contain knowledge of multiple languages without significantly more parameters compared to models trained on only a single language. Our hypothesis is that this could have negative effects on efficient fine-tuning approaches including prefix-tuning. Prefix-tuning in a multilingual setting has already shown to perform well on some tasks [37]. In this thesis we extend the research on multilingual prefix-tuning to a data-to-text task. We compare the method to normal fine-tuning but also introduce a novel way of prefix-tuning called language control prefixes.

# Chapter 2

# Contributions

To efficiently deploy robot journalists based on neural language models in real world scenarios it is important that LMs are easily transferable between languages and domains. Prefix-tuning enables this efficient transfer between domains as shown by Li et al [18]. A solution to efficiently switching between different languages is employing prefix-tuning in combination with a multilingual LM. Zhao et al. [37] introduced multilingual prefix-tuning for some tasks such as natural language inference and question answering. In this thesis, we introduce multilingual prefix-tuning for data-to-text generation and compare it to fine-tuning an LM. If a prefix can be used in combination with different languages, this drastically decreases training time, costs and environmental impact when deploying a data-to-text model. We answer the following research question:

RQ1. **Can prefix-tuning be used as an alternative to fine-tuning an entire multilingual LM for data-to-text generation?**

In this thesis we use mT5 as a multilingual LM. mT5 is a text-to-text language model pre-trained on more than 100 languages and containing more than 580 million parameters. Fine-tuning such a large multilingual LM requires enough data to be available in the target language and task. Domain and language specific data is often limited and expensive to create. An example in the DPG Media case might be creating a model for reporting soccer matches in the Dutch language. Without any existing dataset, this would require DPG Media to manually annotate thousands of samples. Li et al. [18] have shown that prefix-tuning can outperform regular fine-tuning in English low-resource scenarios. In this thesis we experiment with multilingual prefix-tuning in low-resource settings for the data-to-text task. We do this by answering the following research question:

RQ2. **How does prefix-tuning perform compared to fine-tuning on a data-to-text task in multilingual low-resource scenarios?**

Prefix-tuning as introduced by Li et al.[18] does not condition on any sample-specific attributes but uses the same prefix for all samples from a particular dataset. Following on this approach, Clive et al. [18] introduced a prefix that can condition on specific sample attributes. This method, called *control prefixes*, takes a set of categories and values from an input sample and uses those to construct a prefix. An example here might be a data structure describing a women's tennis match. 'Women' and 'tennis' can be attributes to condition the prefix on.

In this thesis we take the concept of control prefixes and modify it to condition on languages instead of sample attributes. We call this method *language control prefixes*. By creating a continuous prefix that is partly language-agnostic and partly conditioned on the processed language we hope to create a single prefix that is able to process multiple languages. Language

control prefixes would allow for more efficient cross-lingual training compared to 'regular' prefix-tuning as introduced by Li et al [18]. This is the case as language control prefixes would require only a single prefix to be trained for multiple languages. Furthermore, language control prefixes might allow for a performance increase in low-resource settings. We hypothesize this because language-specific prefixes from low-resource languages can benefit from a language agnostic prefix that is trained on the same task. We answer the following questions:

RQ3a. **To what extent can language control prefixes be used as a parameter-efficient alternative to general prefix-tuning?**

RQ3b. **To what extent can using language control prefixes lead to an increase in performance in low-resource scenarios compared to general prefix-tuning?**

The different approaches are visualised in Figure 2.1. As can be seen at the top, fine-tuning requires all LM parameters to be updated. In combination with a discrete prefix, multiple tasks and languages can be learned by using a single LM. However, the number of parameters is still two orders of magnitude higher than prefix-tuning. Prefix-tuning is visualised in the second row of Figure 2.1. By training a single prefix for a task-language combination, the Transformer model can be efficiently modified. In combination with a discrete prompt a single prefix can be trained for multiple tasks or languages. This approach is called mixed prefix-tuning and is shown in the third row. Lastly, at the bottom of Figure 2.1, the language control prefixes method is visualised. The prefix is split in a language-agnostic part and a language-specific part. The amount of parameters is similar to prefix-tuning, but becomes lower when only a single language-agnostic prefix is necessary for performing a single task in multiple languages.

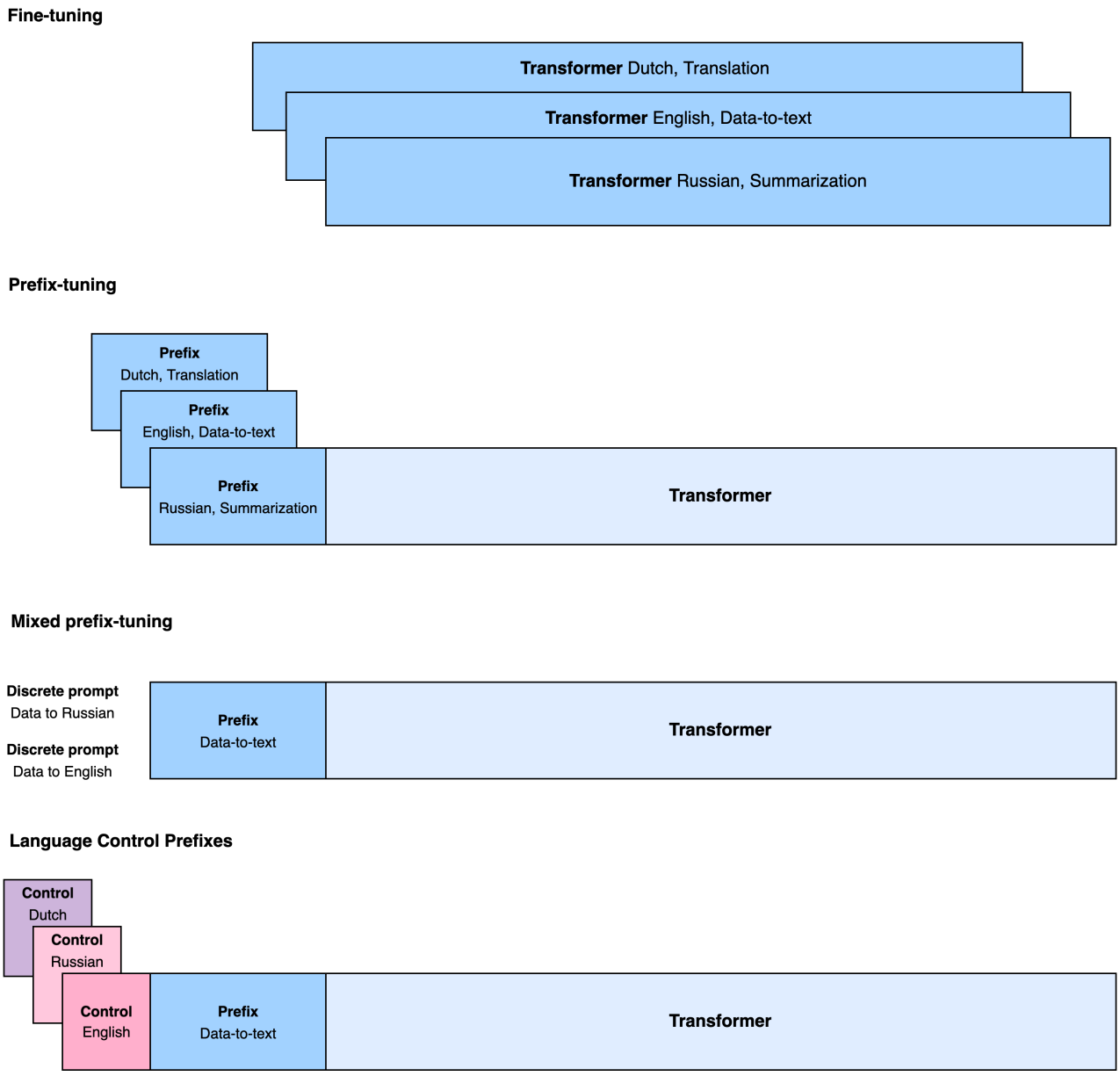


Figure 2.1: Visualisation of four parameter tuning techniques. Firstly there is fine-tuning, where all parameters of a large Transformer LM are updated. Secondly, prefix-tuning updates a task- and language-specific prefix while keeping all LM parameters frozen. Mixed prefix-tuning combines regular prefix-tuning with a discrete prompt, allowing a single prefix to learn a task in multiple languages. Lastly there is the language control prefixes approach, updating both a general task prefix and a language-specific prefix.

# Chapter 3

# Related work

## 3.1 Robot Journalism

In the past decade, software-generated news has slowly found its way into the lives of news editors all around the world. Commonly referred to as 'robot journalism', these pieces of software take structured data and generate a smooth piece of text. Examples of structured data are sports statistics, stock exchange rates or real estate prices [19].

The Washington Post developed their own framework for automated journalism called *Heliograf*. The system first appeared during the 2016 Olympics in Rio de Janeiro. Heliograf got tasked with writing short updates on the Olympics using data from Sports.com. The system was deployed on a liveblog as well as on Twitter and produced multi-sentence updates. Figure 3.1 shows two examples of such tweets. The Washington post expanded their system in the following years. Another example of its application were the 2020 elections where Heliograf was able to give updates on regional results as soon as the data was available. Other big publishers have also implemented AI-journalists in the previous years. Bloomberg has *Cyborg*, Forbes uses *Bertie* and Reuters, the Associated Press and The Guardian all have their own version of an automated reporter as well.

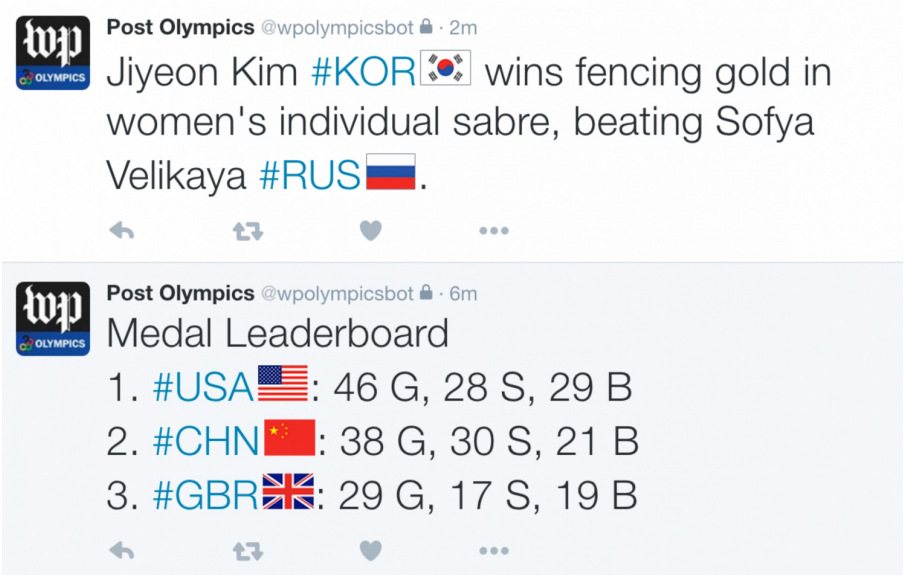


Figure 3.1: Two tweets written by robot reporter *Heliograf* from The Washington Post about the Olympic Games in Rio de Janeiro in 2016.

Fortunately, robot journalism is no longer limited to large publishers. A local newspaper in

the Netherlands, *Dagblad van het Noorden*, recently implemented a system for automated journalism. The software reports on all amateur sports matches in the region using an automated system based on match statistics. A human editor checks the generated piece and optionally adds relevant information before publishing. DPG Media, the company this thesis was written at, is currently experimenting with the automation of writing news articles. DPG Media owned websites like *In de Buurt* provide very local news and could benefit from generating simple articles automatically.

All publishers mentioned in the previous paragraphs have implemented robot journalism in one way or another. To our knowledge all of these implementations use a rule-based approach for generating text. An example of this approach is PASS, a Dutch data-to-text system for soccer [31]. PASS takes data from *Goal.com* and article templates from the MeMo FC corpus [3], a corpus which consists of pairs of soccer reports. Sentences from the MeMo FC corpus are selected and matched to relevant data from Goal.com. After manual editing, the resulting sentences can be used to report on arbitrary soccer matches when linked to the matching statistics. These sentence templates are then combined into a tree structure, resulting in a ready-to-publish news article.

These template-based methods are the current state of the art in production environments at news publishers. The final output of such systems is very predictable and possible errors are easy to fix. Based on qualitative studies, the resulting texts are often very much appreciated by readers [31]. However, templates are extremely limited in their expressivity. Articles often look really alike and use the same structure for their sentences. The only way to diversify in this framework is to include more templates, which is expensive manual work. On top of this, transferring a template database to a new domain often means starting from scratch. For example, almost no sentences from the 'soccer' domain can be used in the context of 'tennis'.

A solution to the problems of limited diversity and expensive manual work is to utilize neural language models. These language models can learn the task of data-to-text generation by taking advantage of the huge amounts of data available in relevant domains. This approach nullifies the need to manually construct sentence templates. Since 2014, various neural encoder-decoder models have been proposed that learn how to map input text to output text [9]. In the next section we introduce the most prominent of these language models and explain the challenges encountered in the process of replacing rule-based methods for robot journalism.

## 3.2 Neural Language Modelling

Language models learn a probability distribution over sequences of words. This way, based on a given sequence of words, the most likely next word can be predicted. Originally, most of these methods were count-based. By simply counting how often words occur together in a certain dataset, one can calculate the probability of certain sequences. More modern approaches use neural networks to compute contextual word embeddings for words and use these to predict the next most likely one.

The recurrent neural network (RNN) was introduced in 1986 by David Rumelhart [29]. This RNN could process sequential data unlike multilayer perceptrons (MLPs) that only work with data where order is of no importance. However, the first RNNs struggled with processing long-term dependencies. In 1997, Hochreiter et al. [13] introduced the LSTM network solving this problem and accelerating the development of sequential networks.

In years that followed, the LSTM was the state of the art for sequential modelling. Tasks like machine translation or text classification were all approached by utilizing an LSTM-related model. In 2017, Vaswani et al. [32] showed that 'attention is all you need' when processing sequential data, introducing the Transformer model. The Transformer uses an attention mechanism to capture the order of the input sequence, and does not need to process the tokens one

by one. For this reason, Transformer models allow for parallel processing of sequences enabling extremely efficient training compared to RNN or LSTM models.

The introduction of the Transformer gave rise a new type of learning for language models. In 2018, Radford et al. [26] introduced the GPT language model. This Generative Pre-trained Transformer is trained without supervision resulting in a general purpose language model. The pre-training happens using a masked language modeling objective. By adapting the model for a specific task after pre-training, the authors were able to achieve state-of-the-art (SOTA) results on tasks ranging from question answering to text classification. The year after, a scaled-up version of the GPT model was released, with a ten-fold increase in both parameters and training data. This model was called GPT-2 [27].

Adapting a pre-trained LM such as GPT-2 for a specific task is called fine-tuning. Fine-tuning a model means that an end-to-end task is performed using the original pre-trained parameters. These parameters of the model are then updated using gradient descent. One can decide to update only a part of the original parameters or introduce some new ones while keeping the original ones frozen. An example of this is prefix-tuning, at the core of thesis. By fine-tuning instead of training a model from scratch, training time and computational requirements are reduced drastically while achieving SOTA performance [26].

After the release of GPT, researchers embraced the approach of training general purpose language representation models. BERT is another example of a highly influential pre-trained Transformer [8] that trains bidirectional representations. Another iteration of this pre-training approach is BART, which focuses on sequence-to-sequence modeling by generalizing the findings of BERT and GPT [17].

The most recent examples of pre-trained language models are GPT-3 and T5 [4] [28]. The model weights behind GPT-3 are not yet made public, keeping us from using the model in this thesis. As Google's T5 model achieves state-of-the-art performance on most language generation tasks, our research focuses on this LM.

## 3.3 Multilingual Language Modelling

The rise of models like GPT and BERT led to researchers pre-training Transformers on different languages than just English. Examples are CamemBERT (French) [21] or BERTje (Dutch) [7]. However, training these large language models requires the availability of enough data and computational resources. For this reason, the introduction of pre-trained LMs has had the undesired effect of limiting the research in NLP to English or other high resource languages [15].

A more general solution towards training models for multiple languages is to pre-train a single model on multiple languages. An example is mBERT [34], a multilingual version of BERT trained on the top 100 languages with the largest Wikipedias. One of the main motivations of training such a multilingual model is to enable transfer learning from one language to another. This concept finds its foundation in the hypothesis that low-resource languages can benefit from high-resource ones due to shared vocabulary, genetic relatedness [22] or contact relatedness [12].

The multilingual version of T5 is called mT5 and was introduced in 2020 [35]. mT5 inherits all characteristics of T5, but is trained on a multilingual variant of T5's dataset, mC4. Multilingual C4 consists of natural text in 101 languages from the Common Crawl web scrape. In this thesis, mT5 is the main language model used for experimentation. In recent months, mT5 has been the standard model for multilingual sequence-to-sequence modelling. As competing models like GPT-3 are not yet openly available, mT5 is the main language model used for experimentation in this thesis.

## 3.4 Lightweight fine-tuning

Fine-tuning pre-trained language models has shown great performance on a wide range of NLP tasks. However, as more downstream tasks are defined, fine-tuning becomes inefficient. For example, in cloud environments it has become easy for anyone to initialize a LM and fine-tune it on a downstream task. As the number of parameters in modern language models is huge, storing a separate copy for each downstream task is very space intensive, resulting in high energy consumption and costs [11]. For this reason, researchers have developed several ways of adapting pre-trained language models for a specific task without the necessity to store and update all parameters.

An example of this parameter-efficient tuning of language models, is adapter tuning [14]. Adapters are modules added between layers of a pre-trained Transformer. These adapters are optimized through gradient descent on a downstream task. In the training process, all original model weights are kept frozen resulting in a factor-10 decrease in learnable parameters. While only adding 3.6% parameters per task compared to fine-tuning the entire model, adapter tuning is able to perform almost as good on a wide variety of tasks [18]. A similar approach to adapter tuning is parameter masking. This method learns what task-specific subset of the LM's parameters should be frozen when fine-tuning [36].

Modern language models like GPT-3 [4] can sometimes even be deployed without any type of fine-tuning. Users can simply prepend a prompt to their input sequence (e.g. TL;DR for summarization) together with a few examples of the task the model is supposed to complete. This process is called prompt-engineering and only requires a manual process of trial-and-error and a negligible amount of space. Although this approach works excellent for numerous tasks, it seems to fail at some tasks like data-to-text [16]. An explanation for this might be that an LM needs to be familiar with the structure of the input. For data-to-text, this input can be of a highly specfic and complex structure.

## 3.5 Prefix-tuning

Another example of lightweight fine-tuning is prefix-tuning. Introduced by Li et al. [18] in 2021, this technique learns a continuous embedding that is prepended to the original input sequence. State-of-the-art language models take a series of input IDs. Each ID stands for a token that is present in the input sequence. The model then encodes this vector of IDs into an embedding that represents meaning in the context of the input sequence. Prefix-tuning learns continuous embeddings that are virtual tokens added to the front of the original token input sequence.

Prefixes are learned without conditioning on any part of the input data. This way, a single prefix is learned for the entire task that needs to be performed. A single weight matrix is updated through gradient descent while performing an end-to-end task. This weight matrix can then generate the prefix. As the weight matrix is the same for the entire task, the prefix stays fixed upon inference time, and no additional computations are necessary to generate it.

The original prefix-tuning paper learns a prefix based on a task. However, there have been attempts to condition the prefix on input-specific data. Clive et al. [6] have introduced control prefixes, allowing a part of the prefix to be conditioned on input-metadata. By introducing a weight matrix that is conditioned on part of the input sequence and adding it to a more general prefix, control prefixes have more control over the generation process.

## 3.6 Prefix-tuning for multilingual models

In the months leading up to this thesis, multiple papers were published that focus on fine-tuning multilingual models using prefixes. Fu et al. [10] introduced PolyPrompt, a framework that uses prompts in combination with a multilingual pre-trained model to perform multiple tasks in multiple languages. This framework completely relies on fine-tuning the original model with discrete tokens as prompt.

Wang et al. [33] introduced Unified Prompt Tuning for Text Classification. This framework tries to capture task-invariant prompting knowledge by learning from a wide variety of tasks. They do this by defining templates and learning what words are used in those templates to optimize performance.

Our approach focuses on optimizing a continuous prefix, rather than learning a discrete prompt. Zhao et al. [37] developed a method that extends on the original prefix-tuning paper [18]. By experimenting with discrete, soft and mixed prompts, the authors conclude that using prompts in multilingual setting outperforms fine-tuning an XLM-RoBERTa model at the natural language inference task. From all recent prefix-related papers, the research from Zhao et al. [37] is closest to our approach. However, we apply prefix-tuning in the context of data-to-text generation instead of natural language inference. On top of this, we introduce language control prefixes as a more effective way of multilingual prefix-tuning.

# Chapter 4

# Prefix-tuning methods

In this section we firstly explain the inner workings of the fine-tuning and prefix-tuning methods that we use in our experiments. Then we introduce language control prefixes as a novel way of learning a multilingual prefix.

## 4.1 Multilingual Fine-tuning

Assume that we have a multilingual language model with an encoder-decoder architecture such as mT5 [35]. All parameters are initialized through the pre-training phase. Fine-tuning this model for the data-to-text task would mean updating the parameters through optimizing a log-likelihood objective using gradient descent:

$$\max_{\phi} \log p_{\phi}(y|x) = \sum_{i \in Y_{\text{idx}}} \log p_{\phi}(z_i|h_{<i})$$

Here $x$ and $y$ are the tokenized input and output sequences respectively. $\phi$ are the model parameters and $Y_{idx}$ are the indices corresponding to the output sequence y. The activation at time step $i$ is defined as $h_i$, computed by the bidirectional Transformer encoder. $z$ is the concatenation of $x$ and $y$. Intuitively this means that that $y$ is predicted autoregressively, conditioned on $x$ and $y$'s left context.

Visualised at the top of Figure 2.1, fine-tuning a Transformer model requires updating all of the pre-trained parameters for each new task and language. Alternatively, a single multilingual LM could be fine-tuned for multiple tasks and languages by using discrete prompts. For example, prepending either 'Translate from Data to English' or 'Translate from Data to Russian' could differentiate between the two different languages inside a single model. Using discrete prompts in combination with task-specific data in a certain language enables steering the output of a language model.

Some recent models learned the meaning of different prompts in their pre-training phase. For example, GPT-3 interprets the prompt 'TL;DR' in such a way that it summarizes the text that follows without fine-tuning [4]. This technique is called zero-shot prompting. Zero-shot prompting is highly model- and prompt-specific and does not apply to our data-to-text generation task. mT5 was not pre-trained on any specific generation task, making it challenging to generate anything sensible in our desired setting without fine-tuning. On top of this, even models that do support zero-shot prompting like GPT-3 have trouble performing the data-to-text task without fine-tuning [18].

## 4.2 Multilingual Prefix-tuning

Discrete prompts are limited in expressivity. While easily understandable for a human, language models might have problems interpreting more complex prompts. Prefix-tuning takes the idea of discrete prompts and casts it to a continuous optimization problem [18]. Instead of using discrete tokens, the prompt exists of continuous embedding vectors. These continuous 'tokens' are propagated through the network's activation functions and influence the processing of subsequent actual tokens in the input sequence. These continuous prefixes are so expressive that optimizing them can achieve SOTA performance on a wide variety of tasks without having to optimize the actual model parameters [18].

For an encoder-decoder model like mT5, prefix-tuning appends continuous prefixes for both the encoder and decoder resulting in $z$ = [Prefix; x; Prefix'; y] [35]. A prefix is generated through a single parameter matrix $P_\theta$. The dimensions of this matrix are $|P_{\text{idx}}| \times \dim(h_i)$. $|P_{\text{idx}}|$ denotes the length of the prefix and $\dim(h_i)$ is the dimensionality of the hidden states. The parameter matrix of the prefix is treated as a multilayer perception (MLP) and optimized using gradient descent while performing the language generation task. During this process, the parameters of the mT5 language model are frozen and only the MLP that generated the prefix embeddings is updated.

As the prefix parameter matrix is not conditioned on any part of the input, a single prefix is trained for each task. In the original prefix-tuning paper [18], this is only done for English tasks. We extend this concept to a multilingual model where a prefix is trained for a specific language and task, similar to the approach taken by Zhao et al. [37]. Training a prefix while keeping all LM parameters frozen only requires updating around 0.1% of the parameters needed for fine-tuning as Table 5.3 shows. Each task-language combination requires learning a separate prefix while keeping the main Transformer model frozen, as visualised in the middle row of Figure 2.1.

## 4.3 Mixed prefix-tuning

By combining the tuned continuous prefix with a discrete prompt, we also investigate the possibility of tuning a single prefix for multiple languages. This mixed prefix-tuning approach does not only prepend a continuous prefix, but also adds a discrete prompt like 'Data to English' to the input sequence. This way, our model can use a single prefix to perform a single task in different languages similarly to using prompts when fine-tuning. For example, tuning a prefix with discrete prompts 'Data to Dutch' and 'Data to English' allows it to handle a task in both languages. By specifying the target language through the discrete part of the prefix, fewer parameters are necessary compared to tuning an new prefix for each task-language combination. Mixed prefix-tuning also enables a prefix to learn language-agnostic information from both languages, possibly increasing performance.

Table 5.3 shows the number of parameters used for mixed prefix-tuning for our particular setup. The middle row of Figure 2.1 shows a general overview of multilingual prefix-tuning. The Transformer model parameters are kept frozen while a prefix is learned for each task-language combination. When using mixed-prefix tuning, the continuous prefix is combined with a discrete prefix to learn

## 4.4 Language control prefixes

Completely new to our approach is the introduction of language control prefixes. Control prefixes were originally introduced by Clive et al. in 2021 [6]. The concept iterates on prefix-tuning

but allows for more fine-grained control. A control prefixes setup always uses a task-specific prefix together with attribute-level parameters conditioned on datapoint-level information. In this section we explain the concept of control prefixes before transferring them to a multilingual setting. As Table 5.3 shows, control prefixes are deployed with a similar number of parameters as prefix-tuning.

A control prefixes setup is very similar to the prefix-tuning approach described in the previous section. All pre-trained language model parameters are frozen and a small MLP is trained that computes a sequence of continuous tokens. These tokens are then prepended to the actual input sequence allowing for a lightweight form of fine-tuning on an NLP task. However, next to a general task prefix $P_\theta$, this approach also introduces a set of control prefixes $C_\theta$ that are sample-specific. An example of such a piece of information could be the news domain of an input sequence, like sport or technology.

### 4.4.1 Control prefixes

For each dataset $\mathcal{Z} = \{\langle X^j, Y^j, G^j \rangle\}_{j=1,\ldots,N}$, $X^j$ is a sample from a specific sequence, $Y^j$ the corresponding label and $G^j$ all information available about the sample. We represent the information as an integer value, defining the category a sample belongs to. The MLP that constructs the control prefix can use this integer value to condition on. Similar to prefix-tuning, the parameters for the general prefix and conditional prefix are optimized through gradient descent.

$$\theta^* = \arg\max_\theta \sum_{j=1}^{N} \log p(Y^j | X^j, G^j; P_\theta, C_\theta, \phi)$$

Here $\theta$ are the parameters of the MLP defining the general prefix ($P$) and control prefixes ($C$). $\phi$ are the frozen parameters defining the large pre-trained language model that is being optimized.

### 4.4.2 Language control prefixes

We introduce the concept of language control prefixes in a multilingual setting. Instead of using information like 'news domain', we condition the prefix on a language. The general part of the prefix can then be used to represent language-agnostic task-specific data. The control prefix can represent language specific information. This means that we now represent our dataset as follows:

$$\mathcal{Z} = \{\langle X^j, Y^j, L^j \rangle\}_{j=1,\ldots,N} X^j$$

Here $X^j$ is a sample from a specific sequence, $Y^j$ the corresponding label and $L^j$ is a categorical integer representing the language of the label.

The bottom of Figure 2.1 shows a visualisation of this approach. The Transformer model on the right has frozen parameters. The prefix in the centre is task-specific and language agnostic. The language control prefixes on the left encode language-specific patterns.

# Chapter 5

# Dataset, metrics and hyperparemeters

In this section we introduce the datasets used, the metrics for evaluation and the hyperparameters for training. All experiments are executed on a system with 61 Gigabytes of memory and an NVIDIA V100 Tensor Core GPU with 16 Gigabytes of memory. As a multilingual sequence-to-sequence model we use mT5 [35]. For the architecture we use the implementation provided by the Hugging Face Transformers library [34]. Specifically we use mT5-base, with pre-trained weights also being provided by the Transformers library [34].

After each epoch of training one of the different methods, we generate output sequences for our development data split. For generation we use the beam search implementation provided by the Transformers library [34]. Beam search generates the next word that is most probable. However, instead of greedily continuing with the selected token, it also keeps track of sequences resulting from choosing one of the other most likely tokens. In our case, we keep track of 5 'beams'. Beam search always finds an output sequence with a higher probability than greedy search, but it is not guaranteed to find an optimal output.

## 5.1 Datasets

The two datasets used to evaluate our multilingual prefix-tuning methods are WebNLG [5] and ToTTo [24]. To quantify the performance of our method on these datasets, we use three metrics that find their origin in machine translation. These are BLEU, METEOR and TER.

### 5.1.1 WebNLG

WebNLG version 3.0 is a bilingual data-to-text dataset. It is available in both English and Russian. The WebNLG dataset maps RDF-triples to text. An RDF triple is a set of three entities formatted as 'subject–predicate–object'. Each sample in the dataset consists of up to seven RDF triples, forming a tree. Each sample has one or several corresponding target generation texts (references) with an average length of 22.5 words. Figure 5.1 shows a snippet of the WebNLG dataset. Notice that it is possible for a single triple set to have multiple reference texts. This is considered upon evaluation time, comparing the generated sentence to all available references.

The dataset was created from raw DBpedia [1] triples. The corresponding references were gathered through crowdsourcing platforms such as Amazing Mechanical Turk. WebNLG consists of samples from 16 different categories such as airport, astronaut, company or celestial body. The amount of samples per data split can be found in Table 5.2. The test and evaluation splits contain both categories that are available in the training set as well as unseen ones. This makes it necessary for the model to learn a function that can extrapolate from only the categories is has seen during training.

The Russian version of WebNLG was created by a machine translation system. The English references were translated automatically before being checked and edited through crowdsourcing. This translation process was only completed for a subset of the entire dataset. As we only use English sentences that are also present in the Russian dataset, the numbers in Table 5.2 are exactly equal.

a. *(John_E_Blaha birthDate 1942_08_26)*
*(John_E_Blaha birthPlace San_Antonio)*
*(John_E_Blaha occupation Fighter_pilot)*

b. *John E Blaha, born in San Antonio on 1942-08-26, worked as a fighter pilot*

Figure 5.1: WebNLG dataset snippet. The three triplets at the top (a) are the input data and the sentence at the bottom (b) is the reference generation.

### 5.1.2 ToTTo

ToTTo is an English data-to-text dataset [24]. The dataset was created by matching Wikipedia tables to noisy descriptions. Then, each cell that is mentioned in the description was highlighted in the table. Matching sentences are rewritten to as clean and varied as possible. Figure 5.2 shows an example from the ToTTo dataset. The final dataset contains 120761 training samples and 7700 development samples with corresponding labels. There is a test set available of 7700 samples that is unlabeled and requires submitting to Google in order to obtain final test results. We use a subset of the complete ToTTo dataset. Exact split sizes can be found in Table 5.2.

ToTTo is a fully English dataset. As our method focuses on a multilingual setting, we choose to automatically translate the target sentences from English to Dutch using a service provided by Amazon Web Services (AWS). AWS Translate provides an easy-to-use API allowing us to translate all training reference sequences to Dutch. Manually inspecting some examples shows that results are close to how a native Dutch speakers would translate them. Table 5.1 shows example translations that show how well the API is able to translate most sequences, but also the noise it introduces.

**Table Title**: Gabriele Becker
**Section Title**: International Competitions
**Table Description**: None

| Year | Competition | Venue | Position | Event | Notes |
|---|---|---|---|---|---|
| **Representing Germany** | | | | | |
| 1992 | World Junior Championships | Seoul, South Korea | 10th (semis) | 100 m | 11.83 |
| 1993 | European Junior Championships | San Sebastián, Spain | 7th | 100 m | 11.74 |
| | | | 3rd | 4x100 m relay | 44.60 |
| 1994 | World Junior Championships | Lisbon, Portugal | 12th (semis) | 100 m | 11.66 (wind: +1.3 m/s) |
| | | | 2nd | 4x100 m relay | 44.78 |
| 1995 | World Championships | Gothenburg, Sweden | 7th (q-finals) | 100 m | 11.54 |
| | | | 3rd | 4x100 m relay | 43.01 |

**Original Text**: After winning the German under-23 100 m title, she was selected to run at the 1995 World Championships in Athletics both individually and in the relay.
**Text after Deletion**: she at the 1995 World Championships in both individually and in the relay.
**Text After Decontextualization**: Gabriele Becker competed at the 1995 World Championships in both individually and in the relay.
**Final Text**: Gabriele Becker competed at the 1995 World Championships both individually and in the relay.

Figure 5.2: ToTTo dataset snippet. The final text at the bottom contains the information highlighted in the table. The table is the input data and the final text at the bottom is the reference generation.

| English source | Dutch translation |
|---|---|
| Prabhu's first directorial film was Chennai 600028. | Prabhu's eerste regiefilm was Chennai 600028. |
| The Proculus was suffect consul in 145, with Decimus Junius Paetus as his colleague. | De Proculus was voldoende consul in 145, met Decimus Junius Paetus als zijn collega. |
| On 2 May, Barcelona defeated Málaga 4–1 at the Camp Nou. | Op 2 mei versloeg Barcelona Málaga 4-1 in Camp Nou. |
| During the 1991 season, Allen rushed for a career-high with 1,036 yards and 8 touchdowns in 18 games, and passed 4,275 yards with 24 touchdowns. | Tijdens het seizoen 1991 haastte Allen zich naar een carrière-high met 1.036 yards en 8 touchdowns in 18 wedstrijden, en passeerde 4.275 yards met 24 touchdowns. |

Table 5.1: Some example translations of English ToTTo sentences translated to Dutch using AWS Translate.

| Dataset | Training | Evaluation | Test |
|---|---|---|---|
| WebNLG English | 14630 | 790 | 1779 |
| WebNLG Russian | 14630 | 790 | 1779 |
| ToTTo English | 15000 | 700 | 700 |
| ToTTo Dutch | 15000 | 700 | 700 |

Table 5.2: Amount of samples for each split from both the WebNLG and the ToTTo dataset.

## 5.2 Metrics

### 5.2.1 BLEU

The Bilingual Evaluation Understudy Score, or BLEU, was introcuced in 2002 by Papineni et al. [23] as a metric for automatic evaluation of machine translation. With over 18.000 citations it is now the most popular metric for sequence-to-sequence generation tasks. BLEU is easy to compute and correlates highly with human evaluation [23]. BLEU captures the precision of a generated sequence compared to a reference. It uses both single words as well as n-grams to do this. The final score is always between 0 and 1 or displayed as a percentage. The higher the score, the better the generated sequence.

### 5.2.2 METEOR

The METEOR (Metric for Evaluation of Translation with Explicit ORdering) score uses unigram precision and recall as a base for calculating generation quality. Banerjee et al. introduced the metric in 2005 [2]. METEOR does not simply compare words from the generation and target sentences, but uses techniques such as stemming and finding synonyms. When overlap has been computed using these techniques, METEOR combines precision, recall and a fragmentation measure to evaluate the order of the generated sentence into a final evaluation score between 0 and 1. The higher the final score, the better.

### 5.2.3 TER

Translation Edit Rate, or TER was introduced by Snover et al. in 2005 [30]. TER measures the amount of edits that a human would have to perform on a generated sentence in order to get one of the target sentences. Similar to BLEU and METEOR, TER produces a score between 0 and 1. However, the score indicates the amount of edits necessary, making a low score better than a high one.

## 5.3 Hyperparameters

### 5.3.1 Fine-tuning

When fine-tuning mT5, we use the following hyperparameters. The model is fine-tuned for 10 epochs. The batch size is set to 4 and the network is updated each 3 steps. As an optimizer AdamW [20] is used, as implemented by Hugging Face [34]. The learning rate is controlled by a scheduler that linearly increases it from 0 to 0.0001 in 1000 steps. After these 1000 steps it linearly decreases to 0 again until training is done. Fine-tuning mT5 requires 582.382.848 parameters to be updated as Table 5.3 shows.

### 5.3.2 Prefix-tuning

The prefix-tuning model generates a prefix length of 10 tokens with an embedding size of 512. The model is trained for 25 epochs. The batch size is set to 4 and the network is updated after each step. As an optimizer we use AdamW [20], implemented by Hugging Face [34]. The learning rate is controlled by a scheduler that linearly decreases it from 0.3 to 0.0 over the entire training time.

### 5.3.3 Language Control Prefixes

The control prefixes model generated a general prefix of length 10, and a language specific prefix of length 1. Each 'token' in this prefix has an embedding size of 512. The model is trained for 25 epochs. The batch size is set to 4 and the network is updated after each step. As an optimizer we use AdamW [20], implemented by Hugging Face [34]. The learning rate is controlled by a scheduler that linearly decreases it from 0.3 to 0.0 over the entire training time.

### 5.3.4 Model size and training time

Table 5.3 shows an overview of the number of parameters that need to be trained for each method. The right column also shows the estimated training time per epoch on our system. As can be seen, prefix-tuning approach requires training only around 0.01% of the parameters required for fine-tuning. On top of this, the training time per epoch is around three times as fast compared to fine-tuning.

| **Method** | **Trainable parameters** | **Training time per epoch** |
|---|---|---|
| Zero-shot | 0 | 0 seconds |
| Fine-tuning | 582.382.848 (for multiple languages) | ±3000 seconds |
| Prefix-tuning | 795.392 (per language) | ±900 seconds |
| Mixed prefix-tuning | 795.392 (for multiple languages) | ±900 seconds |
| Language control prefixes | 797.696 (for multiple languages) | ±900 seconds |

Table 5.3: Amount of trainable parameters for each method and training time per epoch. Each method is combined with a discrete prefix so multiple languages can be used. Only prefix-tuning uses the entire prefix for a single task and language. The training time per epoch is an estimated average based on multiple time measurements.

# Chapter 6

# Experimental evaluation

## 6.1 Comparing prefix-tuning with fine-tuning in a multilingual setting

In this section we compare fine-tuning mT5 to prefix-tuning as a parameter-efficient alternative and answer RQ1: *Can prefix-tuning be used as an alternative to fine-tuning an entire multilingual LM for data-to-text generation?* Firstly, the technical details of both setups are explained. Then, we introduce the different experiments that are performed. Lastly, we discuss the results obtained from our experiments and use them to answer the research question.

### 6.1.1 Fine-tuning as a baseline

The original and most thorough way of optimizing mT5 for generation in a specific language is through fine-tuning all the network parameters. For these fine-tuning experiments we train a separate version of mT5 on WebNLG and ToTTo. We fine-tune mT5 on all available training data from the training set before evaluating on Dutch, Russian and English. Both training and evaluation are performed by prepending a discrete prompt to the input sequence. This way, we can train a single model for the data-to-text task in both available languages in one of the datasets. For all samples we prepend the discrete prompt 'Translate Data to [Language]: ', where [Language] is replaced by either Dutch, English or Russian, depending on the language of the sample being processed.

#### Zero-shot prompting

mT5 is also evaluated on WebNLG and ToTTo without any fine-tuning. We prepend the same discrete prompt as when fine-tuning: 'Translate from Data to [Language]: '. This prompt could guide the model without needing any additional fine-tuning. Zero-shot prompting has shown promising results for modern language models such as T5 [28]. However, compared to T5, mt5 was not trained on any end-to-end task. Our hypothesis would thus be that adding a discrete prefix without any fine-tuning will not show promising results.

### 6.1.2 Prefix-tuning

Prefix-tuning has shown great promise as an efficient alternative to fine-tuning large language models [18]. To show if the performance of prefixes extrapolates to a multilingual data-to-text setting, we implement an model that generates prefixes which can then be fed to mT5.

To construct the prefix, we create a one-dimensional tensor of ascending numbers of a specified prefix length. In our case, the prefix length is 10, resulting in a tensor with numbers

1 to 10. Then we initialize a small neural network to turn these numbers into a continuous prefix.

Firstly, an embedding layer takes the tensor of random numbers. This layer produces an embedding for each ascending number in the original tensor. The embedding size is reliant on the embedding size of the pre-trained LM, so the prefixes can be processed like actual tokens. In our case, mT5 has an embedding size of 512. As directly optimizing this embedding layer yields sub-optimal results [18], we reparametrize the resulting embedding by using a shallow neural net. This network consists of a linear layer, mapping the embedding to a hidden space. After a tanh activation function, another linear layer maps the embedding from the hidden space to the model size again. We decided to go with a hidden layer size of 512, which is the same as the model embedding size. However, any value can be chosen as the size of this hidden dimension.

We firstly focus on implementing prefix-tuning for English using WebNLG. The prefix is trained on the entire English training set without access to any Russian data. By then evaluating on the English development split, we investigate whether or not prefix-tuning is feasible at all when a multilingual sequence-to-sequence model like mT5 is used. This process is repeated for the Russian and Dutch language.

### 6.1.3 Mixed prefix-tuning

To be more parameter efficient, a prefix can be used for both languages per dataset. This also enables the prefix to obtain potential language-agnostic features that can be learned by training on data in multiple languages. By adding a discrete prompt to the input sequence, we experiment with learning a single prefix for multiple languages. This method is called mixed prefix-tuning and is compared against prefix-tuning and fine-tuning.

The setup is very similar to the one introduced for prefix-tuning. The prefix is constructed in exactly the same way. One difference is the fact that we prepend a piece of text to each input sequence. This discrete prompt is the same as the one we use when fine-tuning: 'Translate from Data to [Language]: '. The other difference is that we now train our prefix on all languages from WebNLG and ToTTo at the same time.

### 6.1.4 Results

In this section we show the results to our experiments regarding fine-tuning and (mixed) prefix-tuning. We look at the BLEU evaluation scores over training time and test scores for all introduced evaluation metrics. After looking at the metrics, we qualitatively analyse the generated sentences from each method.

#### Results for English

The left graph in Figure 6.1 shows the evaluation BLEU score over 25 training epochs for the English language. Of all methods, fine-tuning the entire LM increases the performance the fastest. Fine-tuning also converges relatively soon. This might be due to the fact that the scheduler introduces a low learning rate relatively fast compared to the other methods. Prefix-tuning with just English data converges the slowest. This is most likely due to the fact that the other methods can condition on all data in combination with a discrete prompt. This makes it slower for prefix-tuning to keep up in the beginning while just training on English data, seeing fewer samples per epoch, albeit in the language that we evaluate on. The BLEU score of the mixed prefix-tuning approach increases faster compared to the 'normal' prefix-tuning approach, but ends up with lower scores in the long run.

Table 6.1 shows the test performance for all different methods. It becomes instantly clear that zero-shot prompting our model with the prompt 'Translate from Data to English' does not help our pre-trained LM. These results are just as we expected. mT5 is not pre-trained on a particular task, making it necessary for the model to be tuned on at least a few samples before producing useful results. The BLEU score of 1.41 shows the model was able to generate some words or phrases correctly, but we will also show through a qualitative analysis that these results are not properly interpretable.

Similar to what we could see in Figure 6.1, Table 6.1 shows a clear difference in BLEU scores between methods. Prefix-tuning achieves the highest BLEU score. Fine-tuning follows by a margin, a few points higher than mixed prefix-tuning. However, when looking at the METEOR score, results are much closer. The ranking of methods is almost the same as with the BLEU score but there is only a 0.01 difference between methods. Prefix-tuning beats the other methods just slightly. Interesting to note is that for the TER metric, fine-tuning is extremely close to prefix-tuning. Mixed prefix-tuning clearly falls behind accoding to the TER score.

From these results we can conclude that prefix-tuning is an excellent alternative to fine-tuning a multilingual LM for the data-to-text task in English. These findings are similar to the ones done by Li et al [18]. However, we are now working with a multilingual model instead of one that is just pre-trained on English. As this leaves less flexibility for the LM in a particular language, it was not trivial that the performance of prefix-tuning would extend to multilingual models.

Mixed prefix-tuning shows worse performance than its 'normal' variant. This is probably due to the fact that with all English training data available for these experiments, it is beneficial to keep as many parameters as possible available for a specific language. Mixed prefix-tuning mixes two languages in a single prefix, resulting in lower performance but fewer parameters when applied to multiple languages. The approach still comes really close to fine-tuning the entire network.

### Results for Russian

When looking at the Russian evaluation results on the right of Figure 6.1, one thing instantly stands out. Compared to English, for Russian it seems that all parameter-efficient methods do not come close to the performance of fine-tuning the LM. Fine-tuning on all data with the discrete prompt outperforms both prefix-based methods by almost doubling the BLEU score when evaluating on Russian data. This confirms that the findings related to prefix-tuning for the English language do not simply extend to multilingual models. Our results when evaluating on English are really promising for prefix-tuning, but Figure 6.1 shows that this can not just be assumed to work for every language or model.

The conclusions from evaluation BLEU scores are confirmed when looking at the test results in Table 6.2. Fine-tuning achieves the best results for all metrics by a large margin. For METEOR and TER, the difference is not as large as with BLEU, but still clearly visible.

Looking only at the different prefix-tuning methods, results are similar to the ones found when evaluating on English data. Mixed prefix-tuning gets outperformed by prefix-tuning. Especially the difference in BLEU score is relatively large. However, the difference in METEOR is only 0.02 points.

Zero-shot generation using only a discrete prompt on Russian data yields worse results compared to the English evaluation. A BLEU score of 0 suggests that the generated sequences are nowhere close to their references. Again, this result was to be expected as mT5 was not pre-trained on any particular task.

### Results for Dutch

The evaluation BLEU score of the different models on the Dutch ToTTo dataset can be found in the bottom graph of Figure 6.1. Similar to the Russian data, fine-tuning clearly outperforms both variants of prefix-tuning. After just a single epoch of training, fine-tuning the model already doubles the BLEU score of (mixed) prefix-tuning.

Table 6.3 shows similar findings. The test BLEU and TER scores show a clear advantage for fine-tuning. However, just like Figure 6.1, Table 6.3 shows an interesting contrast with English and Russian. This is the fact that mixed prefix-tuning now outperforms prefix-tuning. The time of convergence is also much faster for mixed prefix-tuning compared to its regular variant.

An explanation for the advantage of mixed prefix-tuning could be the increased complexity of ToTTo compared to WebNLG. This complexity implies that to generate correct sentences, the model needs to train on more samples compared to the simpler structure of WebNLG. The extra parameters per language that prefix-tuning offers over mixed prefix-tuning are thus less important than seeing more samples with the ToTTo structure.

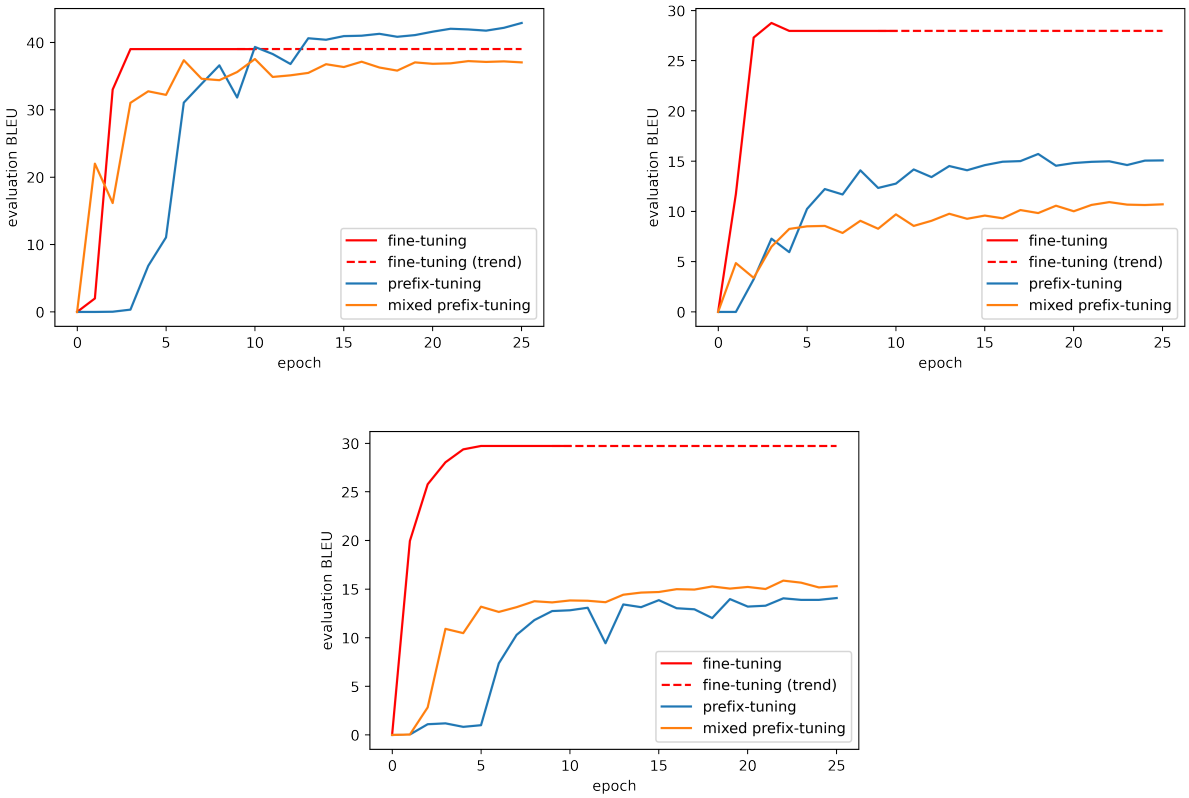


Figure 6.1: BLEU Results for the WebNLG evaluation on English (left), Russian (right) and ToTTo evaluation on Dutch (bottom). All of these methods were trained on all available data.

| Method | N samples | BLEU | METEOR | TER |
|---|---|---|---|---|
| Zero-shot | 0 | 1.41 | 0.04 | 0.97 |
| Fine-tuning | 13211 | 39.00 | 0.32 | 0.5 |
| Mixed prefix-tuning | 13211 | 37.53 | 0.32 | 0.57 |
| Prefix-tuning | 13211 | **42.88** | **0.33** | **0.49** |

Table 6.1: Test set results for the best English models trained on the entire WebNLG dataset

| Method | N samples | BLEU | METEOR | TER |
|---|---|---|---|---|
| Zero-shot | 0 | 0.00 | 0 | 0.99 |
| Fine-tuning | 13211 | **28.00** | **0.22** | **0.65** |
| Mixed prefix-tuning | 13211 | 10.94 | 0.14 | 0.81 |
| Prefix-tuning | 13211 | 15.71 | 0.16 | 0.75 |

Table 6.2: Test set results for the best Russian models trained on the entire WebNLG dataset

| Method | N samples | BLEU | TER |
|---|---|---|---|
| Fine-tuning | 13211 | **29.72** | **0.58** |
| Mixed prefix-tuning | 13211 | 15.66 | 0.74 |
| Prefix-tuning | 13211 | 14.08 | 0.89 |

Table 6.3: Test set results for the best Dutch models trained on the entire ToTTo dataset

### Qualitative analysis

Table 6.4 and Table 6.5 show some examples of generated sentences by the different methods. The reference sentence is what the generated sentences should ideally be similar to.

The first sentence of Table 6.4 has quite a complex structure. Fine-tuning is able to capture part of the sentence, leaving out the part about Juventus being champions. Mixed prefix-tuning actually does include the part about Juventus being Champions, but misunderstands the relationship between the club and league. Prefix-tuning, despite being the top performer based on Table 6.1, also has problems conveying the exact meaning that the reference sentence has. This example clearly shows the difficulties that language models might have with interpreting structured data.

The second sentence in Table 6.1 is an example of prefix-tuning achieving the highest test scores in Table 6.1. The prefix-generated sentence just misses a tiny fraction of the information compared to the other methods. Fine-tuning misses some essential information and mixed prefix-tuning repeats phrases and links statements together which are false.

The final sentence is an example of prefix-tuning not always being the most accurate option. Where all other methods understand the relation between the player and the club, prefix-tuning does not.

Table 6.5 shows three sentences generated from the ToTTo dataset in Dutch. These examples show the difficulty that regular prefix-tuning has with the language and data. For the first sentence, prefix-tuning generates a token that is used upon pre-training time when masking words. Whereas the other methods also struggle with this sentence, prefix-tuning stands out negatively.

The second and third sentences show that fine-tuning is able to generate the sentences that are most complete and closest to the truth. Prefix-tuning misses information or slightly misinterprets it. An example is the confusion between 'gemiddelde' (average) and 'highest' (hoogste) in the second sentence. Mixed prefix-tuning is able to properly interpret most of the meaning, but generates sentences which are grammatically incorrect (sentence 3) or miss information like the °C in sentence 2.

| Method | Generation |
|---|---|
| Reference | A.C. Chievo Verona play in Serie A where Juventus FC have been champions . |
| Fine-tuning | A.C. Chievo Verona play in Serie A. |
| Mixed prefix-tuning | A.C. Chievo Verona is a league where the champions are Juventus FC. |
| Prefix-tuning | The champions of Serie A are Juventus FC and A.C. Chievo Verona. |
| Reference | AS Roma play in Serie A which have Juventus FC as previous champions. AS Roma's ground is in Rome, Italy and they have 70634 members. |
| Fine-tuning | A.S. Roma's ground is Rome, Italy and has 70634 members. |
| Mixed prefix-tuning | The champions of Serie A are Juventus FC who have 70634 members and are located in Rome, Italy. A.S. Roma has 70634 members and the league is known as Serie A. |
| Prefix-tuning | A.S. Roma has 70634 members and play in the Serie A league. Juventus FC is one of the champions. |
| Reference | Rolando Maran plays at the Vicenza Calcio. |
| Fine-tuning | Rolando Maran plays for Vicenza Calcio. |
| Mixed prefix-tuning | Rolando Maran is a player of the Vicenza Calcio club. |
| Prefix-tuning | The Vicenza Calcio club is Rolando Maran. |

Table 6.4: Manually selected generated sentences from the English WebNLG dataset. The models were trained on all English and Russian samples.

| Method | Generation |
|---|---|
| Reference | In 2006 reed Solberg bij de OMV Peugeot Noorwegen de Peugeot 307 WRC in 16 WRC-rondes. |
| Fine-tuning | In 2006 reed Henning Solberg in de Peugeot 307 WRC voor OMV Peugeot Norway. |
| Mixed prefix-tuning | In 2006 won Henning Solberg een Peugeot 307 WRC en een Ford Escort WRC. |
| Prefix-tuning | ¡extra_id_0¿ Peugeot 307 WRC en een Subaru Impreza WRC. |
| Reference | De recordhoge temperatuur in San Diego bedroeg 111° F (44° C) in september. |
| Fine-tuning | De hoogste temperatuur van San Diego was 111° F (44° C) in september. |
| Mixed prefix-tuning | The Climate of San Diego heeft een gemiddeld °F (°C). |
| Prefix-tuning | De gemiddelde temperatuur van San Diego is 111,1 °F. |
| Reference | Idina Menzel werd in 2010 genomineerd voor Choice Music: Group for Glee en in 2014 voor Choice Music: Single. |
| Fine-tuning | Idina Menzel werd genomineerd voor Choice Music Group (2010), Glee (2014) en Choice Music Single (2014). |
| Mixed prefix-tuning | In 2010 won Menzel de Grammy-nominatie voor Choice Music: Group, Glee en Choice Music: Single. |
| Prefix-tuning | In 2014 kreeg Idina Menzel de Teen Choice Awards. |

Table 6.5: Manually selected generated sentences from the Dutch ToTTo dataset. The models were trained on all Dutch and English samples.

### 6.1.5 Discussion

In this section, we discuss RQ1: *Can prefix-tuning be used as an alternative to fine-tuning an entire multilingual LM for data-to-text generation?*.

Firstly, in the English WebNLG setting, prefix-tuning actually outperforms fine-tuning. As prefix-tuning is also a more parameter efficient method, it seems to be a very solid alternative to fine-tuning. This extends the findings from Li et al. [18] to a setting where mT5 is used instead of a single-language model. The qualitative analysis confirms that although the generated results are not perfect, prefix-tuning certainly is not inferior to fine-tuning.

However, looking at the test scores of prefix-tuning for Russian WebNLG and Dutch ToTTo data suggests a different conclusion. Fine-tuning outperforms the more efficient prefix-tuning methods by a large margin. When inspecting some generated sentences from the Dutch dataset, the quality for regular prefix-tuning is confirmed to be much worse than fine-tuning. Mixed prefix-tuning generations are closer to the target sentences than prefix-tuning, which is confirmed by the test metrics. These varying results could be due to the complex nature of the Dutch ToTTo dataset, or to the fact that mT5 could be equipped differently for English generation tasks compared to other languages.

In conclusion, prefix-tuning can definitely be used as a parameter-efficient way of adapting a multilingual model for the data-to-text generation task. However, results may vary widely based on input data structure, choice of multilingual model and output language. Prefix-tuning generations may still be closer to the target than suggested by quantified metrics. Based on our experiments, prefix-tuning should definitely be employed for a WebNLG task in English. For Russian WebNLG-like data, prefix-tuning could be considered after inspecting generation results and deciding on the tradeoff between efficiency and generation accuracy. For the Dutch ToTTo dataset, mixed prefix-tuning would be a preferred over regular prefix-tuning. Based on our qualitative analysis, mixed prefix-tuning is not clearly inferior to fine-tuning.

## 6.2 Prefix-tuning and fine-tuning in low-resource scenarios

The experiments performed in the previous section have shown that prefix-tuning can be used as an alternative to fine-tuning in some multilingual settings. For the English language, performance was really close, whereas prefix-tuning for Russian and Dutch falls behind to regular fine-tuning. In this chapter we introduce similar experiments for low-resource scenarios. Firstly, we explain the different experiments and how we set up fine-tuning and prefix-tuning for these low-resource settings. Then, using the obtained results, we answer RQ2: *How does prefix-tuning perform compared to fine-tuning on a data-to-text task in multilingual low-resource scenarios?*

The experiments for the low-resource settings are performed using both WebNLG and ToTTo. As both datasets are available in two languages, we select one language from each dataset to be low-resource. These low-resource languages will be used for quantitative and qualitative evaluation to answer RQ2. As the authors of this thesis are proficient in English and Dutch, those languages are treated as low-resource. This allows for a proper qualitative analysis of the generation results without the need for a Russian translator. In all WebNLG experiments, Russian is treated as a high-resource language. For ToTTo, English is the high-resource language.

### 6.2.1 Low-resource methods setup

Both fine-tuning and prefix-tuning take place on the entire Russian WebNLG dataset and additionally on a small English part. The amount of English data used is varied to see how

performance changes in low-resource settings. We will experiment with low-resource amounts of 200 and 500. This means that each epoch, mT5 is fine-tuned on all Russian samples and 200 or 500 samples in English. Evaluation is performed on the generation results of the entire English development set.

This process is repeated for the ToTTo dataset, but with Dutch as a low-resource language. For ToTTo we experiment with a Dutch dataset of 200 samples, used in combination with the entire available training set in English. Evaluation is performed on the generation results of the entire Dutch development set.

The setups for both fine-tuning and prefix-tuning are similar to the settings without data restrictions. This includes the use of the discrete prompt to create a mixed prefix approach. The construction of the prefix is done by embedding a tensor of length 10 with an embedding size of 512 and reparametrization is performed to prevent direct optimization as recommended by Li et al [18].

### 6.2.2 Results

#### English WebNLG with 200 samples

When looking at the left graph in Figure 6.2, it becomes clear that fine-tuning does not achieve the highest performance in this setting. After a few epochs of training, the model converges to a BLEU score that's relatively low compared to mixed prefix-tuning. The fact that mT5 has a huge number of parameters probably makes it hard for the model to optimize these efficiently when only 200 samples are available.

Another thing that stands out in Figure 6.2 is the fact that prefix-tuning clearly performs the worst of all methods. Where fine-tuning and mixed prefix-tuning have access to the entire Russian training set, prefix-tuning relies on only 200 English samples. This makes it extremely hard to generalize to the evaluation set, resulting in low BLEU scores, even as training progresses.

Table 6.6 shows that despite a high test BLEU score, fine-tuning actually gets outperformed by mixed prefix-tuning for the METEOR metric. Interesting to note is the extremely low METEOR score of fine-tuning compared its BLEU and TER.

The two method that stand out in this low resource setting is mixed prefix-tuning. The approach is able to efficiently make use of the provided Russian data. On top of this, it has few enough parameters to efficiently optimize for the English language using the available 200 samples.

#### Dutch ToTTo with 200 samples

The graph on the right of Figure 6.2 shows the BLEU evaluation performance for our methods when training ToTTo on 200 Dutch, and all English samples. Compared to the WebNLG setting with 200 samples, it stands out that fine-tuning clearly outperforms mixed prefix-tuning. This might be due to the more complex structure of ToTTo data, requiring more parameters to achieve a high BLEU score. This would explain the difficulties that a prefix with an extremely limited amount of parameters could have, especially in a low-resource setting.

Regular prefix-tuning clearly falls behind, achieving extremely low scores. This is even the case compared to the WebNLG setting, where prefix-tuning scores were already low. The difficulties of (mixed) prefix-tuning in this Dutch ToTTo setting are confirmed when looking at the test results in Table 6.8. For both TER and BLEU, fine-tuning outperforms both prefix-tuning methods.

### English WebNLG with 500 samples

The right of Figure 6.2 shows the evaluation BLEU score for training WebNLG on 500 English samples. The main difference with the graph for 200 samples is the performance of fine-tuning. Similar to the setting with 200 ToTTo samples, mixed prefix-tuning is not able to match the performance of fine-tuning.

Similar to both other settings, is the fact that the line for prefix-tuning stays below all the other methods. However, it increases to a BLEU value above 5, compared to staying around 1 with only 200 samples. Without using any high-resource data, the prefix is not able to properly compete with the other methods.

These findings are confirmed when looking at the optimal models evaluated on the test set in Table 6.7. Interesting to note is the fact that in terms of BLEU, fine-tuning achieves a relatively much higher score compared to the other methods. However, when looking at the METEOR and TER metrics, the results are much closer. Mixed-prefix tuning even outperforms fine-tuning on the METEOR metric. As expected based on the other two settings, regular prefix-tuning performs worst in a low-resource setting with only low-resource samples available.

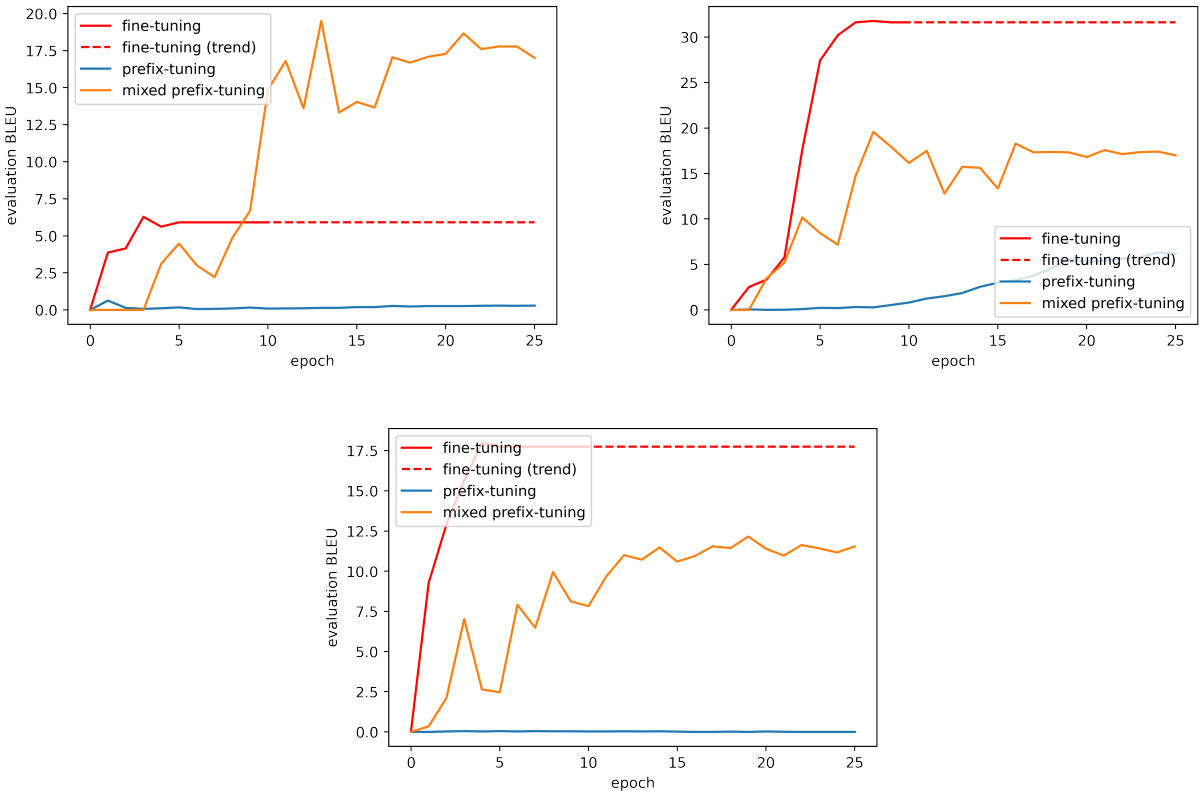


Figure 6.2: BLEU results for the WebNLG evaluation on 200 English samples (left), 500 English WebNLG samples (right) and 200 Dutch ToTTo samples (bottom).

| Method | N samples | BLEU | METEOR | TER |
|---|---|---|---|---|
| Fine-tuning | 200 | 5.91 | 0.07 | 0.82 |
| Mixed prefix-tuning | 200 | **19.50** | **0.21** | **0.75** |
| Prefix-tuning | 200 | 0.63 | 0.05 | 0.99 |

Table 6.6: Test set results for the best models trained on 200 English WebNLG samples

| Method | N samples | BLEU | METEOR | TER |
|---|---|---|---|---|
| Fine-tuning | 500 | **31.77** | 0.13 | **0.76** |
| Mixed prefix-tuning | 500 | 19.58 | **0.19** | 0.77 |
| Prefix-tuning | 500 | 6.32 | 0.14 | 0.83 |

Table 6.7: Test set results for the best models trained on 500 English WebNLG samples

| Method | N samples | BLEU | TER |
|---|---|---|---|
| Fine-tuning | 200 | **17.98** | **0.78** |
| Mixed prefix-tuning | 200 | 12.16 | 0.84 |
| Prefix-tuning | 200 | 0.05 | 0.99 |

Table 6.8: Test set results for the best models trained on 200 Dutch ToTTo samples

### 6.2.3 Qualitative analysis

In this section we qualitatively analyse the same sentences as we did for the high-resource settings from the previous section. Table 6.9, Table 6.10, Table 6.11 show the English WebNLG generation settings with 200 and 500 samples, and the Dutch ToTTo setting with 200 samples.

Table 6.9 firstly shows that regular fine-tuning has difficulties generating sentences in the correct language. Because the model was fine-tuned on mostly Russian and just 200 English samples, the discrete prompt is not enough to clearly signal to the model that the ouput should be generated in English.

Mixed prefix-tuning is better at generating English sentences. However, the structure of these sentences is messy and often incorrect. Phrases like 'numberOfMembers' in the second sentence are literally copied from the input structure without properly generating a phrase. And the same phrase is repeated three times in that same sentence. Relations between entities are also not properly processed, as Chievo is not a champion the Serie A, and Rolando Maran is not a club.

Prefix-tuning on just the 200 provided English samples is clearly not enough to learn the model how to interpret the input structure. The model outputs clear nonsense, something that was already visible in the test metrics in Table 6.6.

Increasing the amount of low-resource data to 500 shows clear benefits for the regular prefix-tuning approach. Table 6.10 shows that instead of producing nonsensical text, prefix-tuning now generates readable text. However, the sentences often still do not make sense in terms of truthfulness. Both fine-tuning and mixed prefix-tuning also still struggle with generating reasonable texts. Mixed prefix-tuning now has the issue of generating sentences in Russian. This might be a unavoidable issue for low-resource setting where the model is also trained on high-resource data. Interesting to note is that fine-tuning generates short sequences, explaining the high scores found in Table 6.7. However, these short sentences are not complete in terms of conveying all meaning form the input data.

Lastly, Table 6.11 shows some Dutch generation results after training on 200 Dutch samples from the ToTTo dataset together will all English samples. The patterns that can be observed are actually really similar to the ones found for the English low-resource setting with 200 samples from Table 6.9. Prefix-tuning generates mostly nonsensical tokens or characters.Despite a significant difference in test scores, mixed prefix-tuning and fine-tuning are not that different based on qualitative analysis of these samples. Most relevant information can be found in the sequences from mixed prefix-tuning, despite fine-tuning showing a better grammatical quality.

| Method | Generation |
|---|---|
| Reference | A.C. Chievo Verona play in Serie A where Juventus FC have been champions . |
| Fine-tuning | Russian text. Find exact generation in Figure A.1. |
| Mixed prefix-tuning | Serie "A" is a league where Chievo is a champion. |
| Prefix-tuning | extra_id_0. extra_id_1..... (continues...) |
| Reference | AS Roma play in Serie A which have Juventus FC as previous champions. AS Roma's ground is in Rome, Italy and they have 70634 members. |
| Fine-tuning | Russian text. Find exact generation in Figure A.1. |
| Mixed prefix-tuning | "Roma" has numberOfMembers 70634. The team has numberOfMembers 70634. The team has numberOfMembers 70634. |
| Prefix-tuning | extra_id_0 A.S_Roma extra_id_1 extra_id_2. (continues...) |
| Reference | Rolando Maran plays at the Vicenza Calcio. |
| Fine-tuning | Russian text. Find exact generation in Figure A.1. |
| Mixed prefix-tuning | Rolando Maran is a club in Vicenza. |
| Prefix-tuning | extra_id_0/ / / / / / / / / / /. (continues...) |

Table 6.9: Manually selected generated sentences from the English WebNLG dataset. The models were trained on 200 English samples and all available Russian samples.

| Method | Generation |
|---|---|
| Reference | A.C. Chievo Verona play in Serie A where Juventus FC have been champions . |
| Fine-tuning | A. |
| Mixed prefix-tuning | - champion of Chievo Verona. |
| Prefix-tuning | Juventus Football Verona is a Serie A. C. Chievo Verona. |
| Reference | AS Roma play in Serie A which have Juventus FC as previous champions. AS Roma's ground is in Rome, Italy and they have 70634 members. |
| Fine-tuning | The league of A. |
| Mixed prefix-tuning | Russian text. Find exact generation in Figure A.1. |
| Prefix-tuning | A.S._Roma is a Serie A's champions. |
| Reference | Rolando Maran plays at the Vicenza Calcio. |
| Fine-tuning | The Vicenza Calcio. |
| Mixed prefix-tuning | Russian text. Find exact generation in Figure A.1. |
| Prefix-tuning | Rolando Maran's club is Vicenza Calcio. |

Table 6.10: Manually selected generated sentences from the English WebNLG dataset. The models were trained on 500 English samples and all available Russian samples.

| Method | Generation |
|---|---|
| Reference | In 2006 reed Solberg bij de OMV Peugeot Noorwegen de Peugeot 307 WRC in 16 WRC-rondes. |
| Fine-tuning | In 2006 werd Henning Solberg de Peugeot 307 WRC in OMV Peugeot Norway. |
| Mixed prefix-tuning | Peugeot rijdt in 2006 met een Peugeot 307 WRC en een Peugeot 307 WRC. |
| Prefix-tuning | extra_id_0. extra_id_1..... (continues...) |
| Reference | De recordhoge temperatuur in San Diego bedroeg 111° F (44° C) in september. |
| Fine-tuning | The gemiddelde temperatuur van San Diego is 111 °F (44 °C). |
| Mixed prefix-tuning | The gemiddelde temperatuur in San Diego is 111 °F (°C). |
| Prefix-tuning | extra_id_0 Xbox Xbox Xbox (continues...) |
| Reference | Idina Menzel werd in 2010 genomineerd voor Choice Music: Group for Glee en in 2014 voor Choice Music: Single. |
| Fine-tuning | Idina Menzel won twee awards voor Choice Music Group (2010), Glee (2014) en Choice Music Single (2014). |
| Mixed prefix-tuning | In 2010, Idina Menzel kreeg de Award voor Choice Music: Group en 2014 voor Choice Music: Single. |
| Prefix-tuning | extra_id_0 Idina Menzel: extra_id_1 (continues...) |

Table 6.11: Manually selected generated sentences from the Dutch ToTTo dataset. The models were trained on 200 Dutch samples and all available English samples.

### 6.2.4 Discussion

In this section, we discuss RQ2: *How does prefix-tuning perform compared to fine-tuning on a data-to-text task in multilingual low-resource scenarios?*.

Firstly, regular prefix-tuning struggles greatly in low-resource settings as it has no access to an alternative high-resource language. This method can not be used as an alternative to fine-tuning.

Learning a prefix with both a high and low-resource language through mixed prefix-tuning shows much more promising results. In the English setting with 200 samples, mixed prefix-tuning is able to outperform fine-tuning by a large margin. Qualitative analysis confirms that mixed prefix-tuning is the only method able to generate proper sentences in this setting. However, when applied to different settings, fine-tuning takes over as the method with the highest test scores for most metrics. These differences are often rather small, especially for the English WebNLG task, and not clearly visible through the qualitative analysis of generated sequences.

We conclude that mixed prefix-tuning is not a guarantee for outperforming regular fine-tuning. However, the method speeds up training time and needs only 0.01% of the parameters required for fine-tuning. Given the fact that for some settings performance of mixed prefix-tuning is similar to fine-tuning or even better, it should be considered as an efficient alternative to fine-tuning in low-resource settings.

## 6.3 Language control prefixes for more effective multilingual prefix-tuning

In the previous sections we introduced (mixed) prefixes in a multilingual setting. Then we took those approaches to a low resource setting. In this section, we introduce language control prefixes. These type of prefixes partly condition on the current language being processed. This language-specific part is combined with a language-agnostic part, allowing for a clearer separation between language and task. Intuitively, the language specific part is similar to a discrete prompt introduced in the previous sections. However, by optimizing the prompt as a continuous prefix, we hope to give the model more control over its generations.

In this section, we introduce the technical setup of language control prefixes. We then test the control prefixes performance on all tasks previously performed. Using the obtained results, we answer RQ3a: *To what extent can language control prefixes be used as a parameter-efficient alternative to general prefix-tuning?* and RQ3b: *To what extent can using language control prefixes lead to an increase in performance in low-resource scenarios compared to general prefix-tuning?*

### 6.3.1 Language control prefixes setup

The architecture that generates a control prefix is very similar to one used for prefix-tuning. Control prefixes uses a tiny fraction more parameters (797.696) compared to prefix-tuning (795.392), as Table 5.3 shows.

Just like with prefix-tuning, control prefixes first initializes a tensor containing an ascending range of numbers from 1 to the prefix length. This tensor will be used to construct the general part of the prefix. The control prefix does not require any sort of initialization, as it uses an integer defined by the language to condition on.

The general prefix and control prefixes information both have their own embedding layer. These layers map the one-dimensional tensors to an embedding of size 512, the embedding size of mT5. These embeddings are then concatenated, to create a final prefix combining the general information of the task at hand with language-specific information from the control prefix. Again, directly optimizing these embeddings is sub-optimal, so an MLP is defined to reparametrize the total prefix. The MLP again consists of a hidden layer mapping to an embedding size of 512, a tanh activation function and a final linear layer creating the embedding that is fed to the language model. Figure 6.3 shows a simplified overview of the model that creates the language control prefixes.

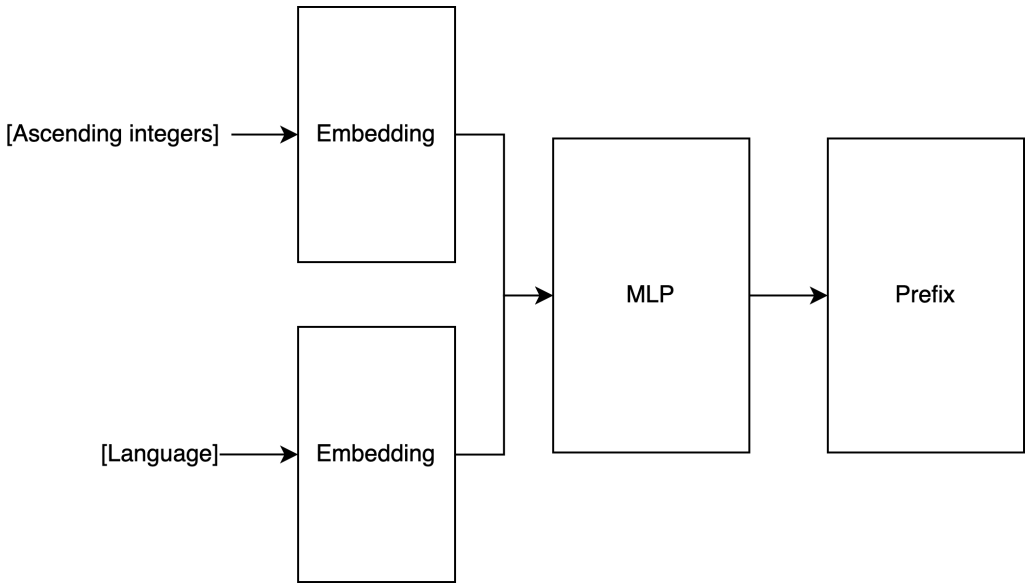


Figure 6.3: Simplified representation of the language control prefixes model. One embedding is conditioned on the language of the sample that is processed. The other embedding is created by the same ascending integers for each sample. These embeddings are concatenated and then reparametrized by an MLP.

We perform the language control prefixes experiments using both WebNLG and ToTTo. To properly compare the performance to the experiments from the previous sections, we introduce both low-resource and high-resource scenarios. For WebNLG we train the model on all available Russian samples and 200, 500 or all English samples. For ToTTo, we only have two setups, one with all Dutch samples and one with 200. The ToTTo setup always has access to all English samples.

### 6.3.2 Results

The graphs shown below are the same as the ones introduced in the previous sections, with the addition of the language control prefixes setup. The same goes for the different tables, where all greyed out results are from previous sections. The bottom row shows the results introduced in this section.

#### Language control prefixes in a high-resource setting

Figure 6.4 shows the three high-resource settings in which we experimented with language control prefixes. The English and Russian ones (left and right) are both based on WebNLG data, whereas the Dutch graph (bottom) is from the ToTTo dataset.

For both English and Russian, control prefixes has a similar evaluation BLEU score to normal prefix-tuning. This is noteworthy, because normal prefix-tuning requires a dedicated prefix for a single language, whereas control prefixes can condition on multiple ones. Mixed-prefix tuning struggled with representing data from multiple languages in a single prefix, resulting in worse performance compared to regular prefix-tuning.

It was to be expected that with all data available, control prefixes would perform similar to mixed prefix-tuning. However, the continuous optimizable conditional prompt seems to outperform a predefined discrete prompt. Using control prefixes is more parameter-efficient than training a separate prefix for each language.

For Dutch ToTTo data, control prefixes seems to underperform compared to the other prefix-tuning methods. The added complexity of a conditional prefix in combination with a

more complex dataset actually causes the model to learn barely anything until after 12 epochs of training. The predefined prompt of mixed prefix-tuning helps the model in learning the data structure almost instantly before gradually increasing.

The findings from the evaluation BLEU scores in Figure 6.4 are confirmed when looking at the test scores in Table 6.12, Table 6.13 and Table 6.14. In the WebNLG setting for English, control-prefix is on a similar level as prefix-tuning, even outperforming the method when it comes to the METEOR score.

For the Russian WebNLG setting described in Table 6.12, language control prefixes clearly gets outperformed by fine-tuning. However, the performance is again very close to that of prefix-tuning. For all three metrics, the differences are extremely small. One should consider here that control prefixes can scale to multiple languages more efficiently, requiring fewer parameters per language.

Table 6.14 shows that for this more complex task in Dutch, control prefixes is not able to match the performance of fine-tuning and other prefix-tuning methods for most metrics. Interesting to note is that control prefixes does outperform regular prefix-tuning for the TER metric. However, mixed prefix-tuning beats both other methods on all three metrics.

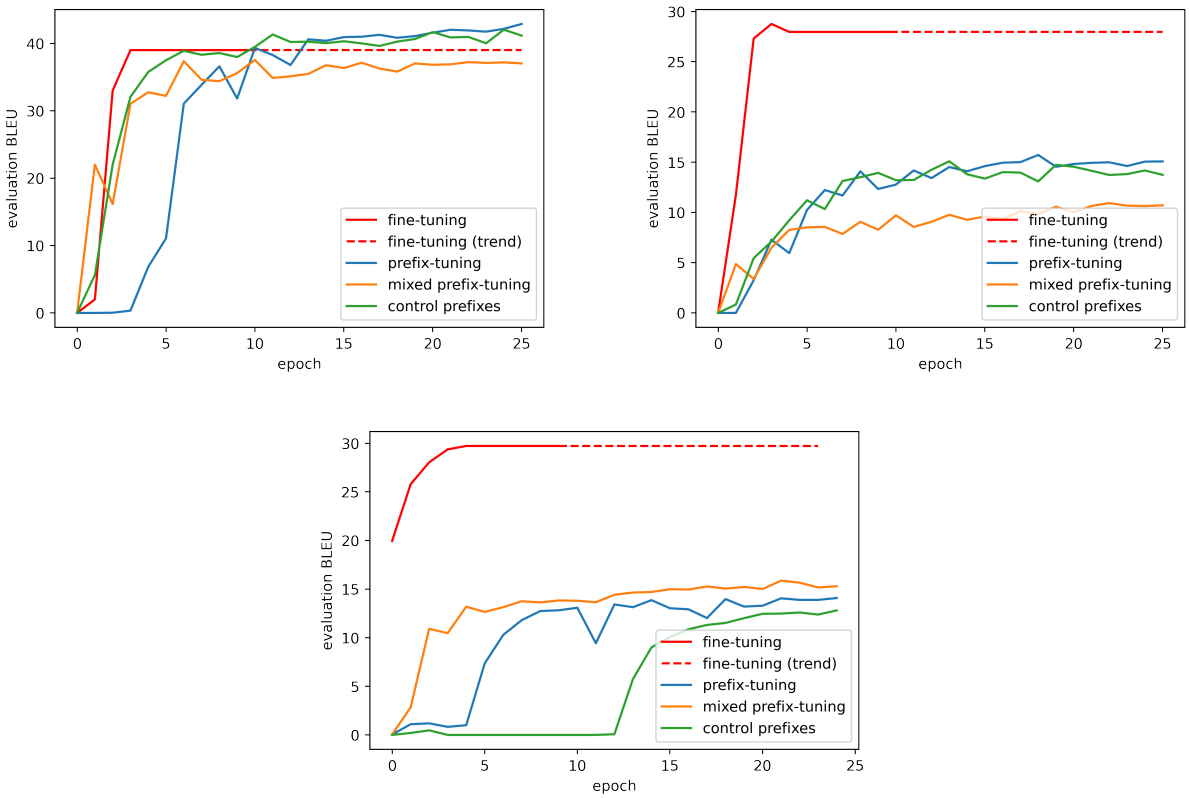


Figure 6.4: BLEU Results for the WebNLG evaluation on English (left), Russian (right) and ToTTo evaluation on Dutch (bottom). All of these methods were trained on all available data.

| Method | N samples | BLEU | METEOR | TER |
|---|---|---|---|---|
| Zero-shot | 0 | 1.41 | 0.04 | 0.97 |
| Fine-tuning | 13211 | 39.00 | 0.32 | 0.5 |
| Mixed prefix-tuning | 13211 | 37.53 | 0.32 | 0.57 |
| Prefix-tuning | 13211 | **42.88** | 0.33 | **0.49** |
| Control prefixes | 13211 | 42.09 | **0.34** | 0.51 |

Table 6.12: Test set results for the best English models trained on the entire WebNLG dataset

| Method | N samples | BLEU | METEOR | TER |
|---|---|---|---|---|
| Zero-shot | 0 | 0.00 | 0 | 0.99 |
| Fine-tuning | 13211 | **28.00** | **0.22** | **0.65** |
| Mixed prefix-tuning | 13211 | 10.94 | 0.14 | 0.81 |
| Prefix-tuning | 13211 | 15.71 | 0.16 | 0.75 |
| Control prefixes | 13211 | 14.76 | 0.16 | 0.73 |

Table 6.13: Test set results for the best Russian models trained on the entire WebNLG dataset

| Method | N samples | BLEU | TER |
|---|---|---|---|
| Fine-tuning | 13211 | **29.72** | **0.58** |
| Mixed prefix-tuning | 13211 | 15.66 | 0.74 |
| Prefix-tuning | 13211 | 14.08 | 0.89 |
| Control prefixes | 13211 | 12.38 | 0.85 |

Table 6.14: Test set results for the best Dutch models trained on the entire ToTTo dataset

### Language control prefixes in a low-resource setting

Figure 6.5 shows that for all experimental settings, control prefixes follows a similar trajectory as mixed prefix-tuning. For both English WebNLG settings, control prefixes outperforms mixed prefix-tuning clearly. As more data becomes available in the scenario with 500 samples, this difference becomes bigger.

Similar to what we saw in the high-resource setting is the fact that control prefixes seem to struggle with the ToTTo dataset in Dutch. As the structure of this dataset is more complex, learning a continuous conditional prefix might be harder than just using a discrete prompt.

These findings are also reflected in Table 6.15, Table 6.16 and Table 6.17. The test result for English WebNLG data show language control prefixes as the best performer over mixed prefix-tuning. In Table 6.15, only the METEOR metric is the same for both of these methods. TER and BLEU show a clear advantage for control prefixes. As Table 6.16 shows, increasing the amount of available data also shows a difference for the METEOR metric.

The test results on Dutch ToTTo in Table 6.17 show that control prefixes does not come close to the performance of fine-tuning. Both TER and BLEU are also lower compared to mixed prefix-tuning, confirming what we observed in Figure 6.5.

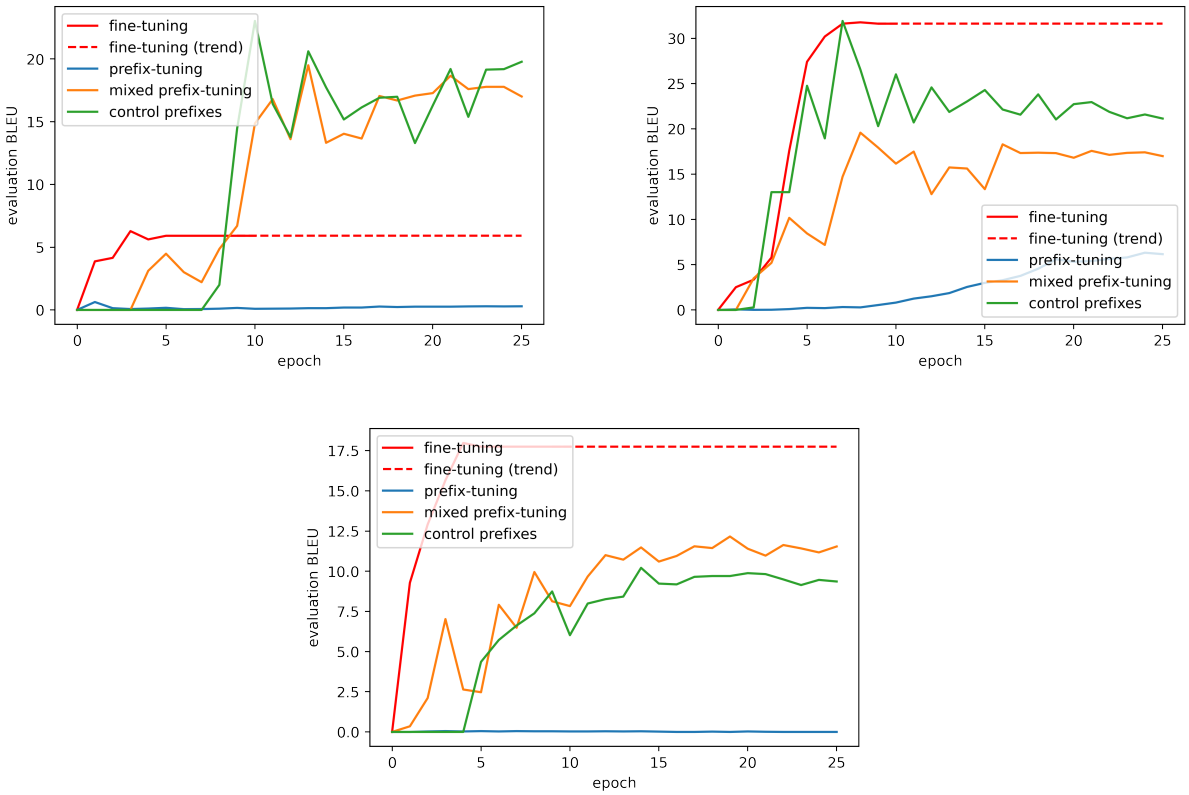


Figure 6.5: BLEU results for the WebNLG evaluation on 200 English samples (left), 500 English WebNLG samples (right) and 200 Dutch ToTTo samples (bottom).

| Method | N samples | BLEU | METEOR | TER |
|---|---|---|---|---|
| Fine-tuning | 200 | 5.91 | 0.07 | 0.82 |
| Mixed prefix-tuning | 200 | 19.50 | 0.21 | 0.75 |
| Prefix-tuning | 200 | 0.63 | 0.05 | 0.99 |
| Control prefixes | 200 | **23.03** | **0.21** | **0.70** |

Table 6.15: Test set results for the best models trained on 200 English WebNLG samples

| Method | N samples | BLEU | METEOR | TER |
|---|---|---|---|---|
| Fine-tuning | 500 | 31.77 | 0.13 | 0.76 |
| Mixed prefix-tuning | 500 | 19.58 | 0.19 | 0.77 |
| Prefix-tuning | 500 | 6.32 | 0.14 | 0.83 |
| Control prefixes | 500 | **31.97** | **0.20** | **0.68** |

Table 6.16: Test set results for the best models trained on 500 English WebNLG samples

| Method | N samples | BLEU | TER |
|---|---|---|---|
| Fine-tuning | 200 | **17.98** | **0.78** |
| Mixed prefix-tuning | 200 | 12.16 | 0.84 |
| Prefix-tuning | 200 | 0.05 | 0.99 |
| Control prefixes | 500 | 10.21 | 0.95 |

Table 6.17: Test set results for the best models trained on 200 Dutch ToTTo samples

### 6.3.3 Qualitative analysis

Similar to our (mixed) prefix-tuning approaches in all different settings, we also perform a qualitative analysis of the language control prefixes setup. Table 6.18 and Table 6.19 show these results.

Table 6.18 displays that language control prefixes are able to generate sensible sequences in all different WebNLG settings. However, the fewer amount of samples the model has available, the less grammatically correct the output sequences become. For example, results have small mistakes like 'is' instead of 'are'. Interesting to see is that for the final sentence, the model generates a '-' instead of a proper sentence. The '-' still portrays the meaning of the sequence correctly, but is not what is expected from such a model.

The Dutch ToTTo generations showed in Table 6.19 have a similar pattern as the English ones. All sequences are sensible without showing masking tokens or changes in language like low-resouce settings from the other methods do. Interesting to note is that for these three particular sentences, the difference in quality between the low- and high-resource settings is almost nonexistent.

| **Setting** | **Generation** |
|---|---|
| Target | A.C. Chievo Verona play in Serie A where Juventus FC have been champions. |
| WebNLG English | A.C. Chievo Verona is the champion of Serie A. |
| WebNLG English 200 | Fiorentina FC Chievo Verona is champions in Serie A. |
| WebNLG English 500 | Chievo Verona champions of the Serie A. |
| Target | AS Roma play in Serie A which have Juventus FC as previous champions. AS Roma's ground is in Rome, Italy and they have 70634 members. |
| WebNLG English | A.S. Roma's ground is Rome, Italy and have 70634 members. The champions are Juventus F.C. |
| WebNLG English 200 | Serie A (Serie A) - champions of Serie A. |
| WebNLG English 500 | a league of the Serie A is located in Rome, Italy. |
| Target | Rolando Maran plays at the Vicenza Calcio. |
| WebNLG English | Vicenza Calcio is the club of Rolando Maran. |
| WebNLG English 200 | Rolando Maran - Vicenza Calcio. |
| WebNLG English 500 | Rolando Mara's club is Vicenza Calcio. |

Table 6.18: Manually selected generated sentences from language control prefix generations on the English WebNLG dataset. The models were trained on either all, 200 or 500 English samples and all Russian samples.

| Setting | Generation |
|---|---|
| Target | In 2006 reed Solberg bij de OMV Peugeot Noorwegen de Peugeot 307 WRC in 16 WRC-rondes. |
| ToTTo Dutch | OMV Peugeot Norway. Henning Solberg heeft een Peugeot 307 WRC en een Subaru Impreza WRC. Henning Solberg heeft een Peugeot 307 WRC. |
| ToTTo Dutch 200 | In 2006 debuteerde Henning Solberg in Peugeot 307 WRC. |
| Target | De recordhoge temperatuur in San Diego bedroeg 111° F (44° C) in september. |
| ToTTo Dutch | The gemiddelde temperatuur in San Diego is 111,44 graden. |
| ToTTo Dutch 200 | The gemiddelde temperatuur in San Diego is °F (°F) and °F (°F). |
| Target | Idina Menzel werd in 2010 genomineerd voor Choice Music: Group for Glee en in 2014 voor Choice Music: Single. |
| ToTTo Dutch | In 2010 won Idina Menzel de Teen Choice Awards. |
| ToTTo Dutch 200 | In 2010 kreeg Idina Menzel een nomination voor de Teen Choice Awards. |

Table 6.19: Manually selected generated sentences from language control prefix generations on the Dutch ToTTo dataset. The models were trained on either all or 200 Dutch samples and all available English samples.

### 6.3.4 Discussion

In this section, we discuss RQ3a: *To what extent can using language control prefixes lead to an increase in performance in low-resource scenarios compared to general prefix-tuning?*. Then we answer RQ3b: *To what extent can using language control prefixes lead to an increase in performance in low-resource scenarios compared to general prefix-tuning?*.

To answer RQ3a, language control prefixes can be used as a parameter-efficient alternative to general prefix-tuning. Language control prefixes outperforms all other prefix-tuning methods in 2 of our 3 high-resource settings in terms of test metrics. Manual inspection of generated sentences shows a similar conclusion, with all sentences being at least sensible, albeit not always fully truthful. An exception is the Dutch ToTTo setting where control prefixes performs worse than all other prefix-tuning methods. However, the qualitative analysis of the results shows that differences are really small in terms of generation quality and training for more epochs with more available parameters might close the gap.

Language control prefixes show even more potential in low-resource settings compared to high-resource ones. To answer RQ3b, we observed that control prefixes outperform all other methods including fine-tuning and mixed prefix-tuning in English WebNLG tasks. Only in the Dutch ToTTo setting, regular fine-tuning is able to achieve higher test scores with control prefixes being around the same performance as mixed prefix-tuning. In short, control prefixes leads to an increase in performance in most low-resource scenarios compared to other prefix-tuning methods.

# Chapter 7

# Conclusion

Prefix-tuning is an extremely parameter efficient way of modifying a language model for the data-to-text generation task. Especially in low-resource settings, the compact representations of these prefixes are able to perform well. Control prefixes allow regular prefixes to be conditioned on attribute level data, allowing for a potential increase in performance.

In this work we took prefix-tuning for data-to-text to a multilingual setting. We answered RQ1: *Can prefix-tuning be used as an alternative to fine-tuning an entire multilingual LM for data-to-text generation?*, by showing that in some settings the performance of prefix-tuning is very close to regular fine-tuning while being extremely parameter efficient. However, the performance of prefix-tuning is highly dependent on the exact setting in which it is deployed.

Prefix-tuning in the English data-to-text scenario shows most promising results in low-resource scenarios. We investigated this in a multilingual setting by answering RQ2: *How does prefix-tuning perform compared to fine-tuning on a data-to-text task in multilingual low-resource scenarios?*. Mixed prefix-tuning was able to efficiently profit from an available high-resource language and can be used as a parameter-efficient alternative to fine-tuning despite lacking superior test results in some settings.

Our main contribution is the introduction of language control prefixes. By answering RQ3a: *To what extent can language control prefixes be used as a parameter-efficient alternative to general prefix-tuning?*, we showed that language control prefixes outperform other prefix-tuning methods in most settings. This finding was extended by answering RQ3b: *To what extent can using language control prefixes lead to an increase in performance in low-resource scenarios compared to general prefix-tuning?*. Control prefixes outperformed all other methods in most low-resource scenarios, showing the potential of the method for multilingual data-to-text tasks.

In short, prefix-tuning is a highly effective way of efficiently adapting a pre-trained language model for a multilingual data-to-text task. Especially in low resource settings this becomes apparent. The introduction of language control prefixes takes this a step further by outperforming all other methods in most low-resource settings.

Experimenting with language control prefixes in more settings and languages is left for future work. By combining such research with a thorough process of hyperparameter tuning, the impact of control prefixes on efficient mulilingual prefix-tuning could be further analysed. Another open question is why exactly the performance of prefix-tuning is dependent on the setting where it is applied. Future research could look into language features or model limitations that may influence the performance of prefix-tuning methods. Lastly, we propose a research direction of more advanced prefix construction. While control prefixes already condition on part of an input sample, more thorough approaches could be constructed. An example would be to represent the input structure as a graph and use graph-based neural networks to compute the final prefix.

# Appendix A

# Appendix

1. ФК "Кьево"(Верона) играет в Чемпионате Италии по футболу, где ФК "Ювентус"были чемпионами.
2. Домашняя площадка ФК "Рома"находится в Риме, Италия. Они играют в Чемпионате Италии по футболу и имеют 70634 участника.
3. Роландо Маран играет за Виченцу.
4. ФК Рома имеет 70634 членов и является чемпионом Италии по футболу. Команда А.С. Рома имеет 70634 членов и имеет номер 70634 членов.
5. Аргентинский футбольный клуб Ролано Маран является членом Висенса Calcio.

Figure A.1: Russian sentences generated when evaluating different prefix-tuning methods.

# Bibliography


[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007.

[2] Satanjeev Banerjee and Alon 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*, pages 65–72, 2005.

[3] Nadine Braun, Martijn Goudbeek, and Emiel Krahmer. The multilingual affective soccer corpus (masc): Compiling a biased parallel corpus on soccer reportage in english, german and dutch. In *Proceedings of the 9th International Natural Language Generation Conference*, pages 74–78, 2016.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Thiago Castro Ferreira, Claire Gardent, Nikolai Ilinykh, Chris van der Lee, Simon Mille, Diego Moussallem, and Anastasia 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+)*, pages 55–76, Dublin, Ireland (Virtual), 12 2020. Association for Computational Linguistics.

[6] Jordan Clive, Kris Cao, and Marek Rei. Control prefixes for text generation. *arXiv preprint arXiv:2110.08329*, 2021.

[7] Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. Bertje: A dutch bert model. *arXiv preprint arXiv:1912.09582*, 2019.

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

[9] Chenhe Dong, Yinghui Li, Haifan Gong, Miaoxin Chen, Junxin Li, Ying Shen, and Min Yang. A survey of natural language generation. *arXiv preprint arXiv:2112.11739*, 2021.

[10] Jinlan Fu, See-Kiong Ng, and Pengfei Liu. Polyglot prompt: Multilingual multitask promptraining. *arXiv preprint arXiv:2204.14264*, 2022.

[11] Stefanos Georgiou, Maria Kechagia, Tushar Sharma, Federica Sarro, and Ying Zou. Green ai: Do deep learning frameworks have different costs? ACM: Association for Computing Machinery, 2022.

[12] Vikrant Goyal, Anoop Kunchukuttan, Rahul Kejriwal, Siddharth Jain, and Amit Bhagwat. Contact relatedness can help improve multilingual NMT: Microsoft STCI-MT @ WMT20. In *Proceedings of the Fifth Conference on Machine Translation*, pages 202–206, Online, November 2020. Association for Computational Linguistics.

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

[14] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[15] Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. The state and fate of linguistic diversity and inclusion in the nlp world. *arXiv preprint arXiv:2004.09095*, 2020.

[16] Moniba Keymanesh, Adrian Benton, and Mark Dredze. What makes data-to-text generation hard for pretrained language models? *arXiv preprint arXiv:2205.11505*, 2022.

[17] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[18] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online, August 2021. Association for Computational Linguistics.

[19] Carl-Gustav Linden. Decades of automation in the newsroom: Why are there still so many jobs in journalism? *Digital journalism*, 5(2):123–140, 2017.

[20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[21] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de La Clergerie, Djamé Seddah, and Benoît Sagot. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*, 2019.

[22] Toan Q. Nguyen and David Chiang. Transfer learning across low-resource, related languages for neural machine translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 296–301, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing.

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

[24] Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of EMNLP*, 2020.

[25] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online, April 2021. Association for Computational Linguistics.

[26] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

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

[28] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

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

[30] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John 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*, pages 223–231, 2006.

[31] Chris van der Lee, Emiel Krahmer, and Sander Wubben. Pass: A dutch data-to-text system for soccer, targeted towards specific audiences. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 95–104, 2017.

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

[33] Jianing Wang, Chengyu Wang, Fuli Luo, Chuanqi Tan, Minghui Qiu, Fei Yang, Qiuhui Shi, Songfang Huang, and Ming Gao. Towards unified prompt tuning for few-shot text classification. *arXiv preprint arXiv:2205.05313*, 2022.

[34] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[35] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online, June 2021. Association for Computational Linguistics.

[36] Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. Masking as an efficient alternative to finetuning for pretrained language models. In *EMNLP 2020*, 2020.

[37] Mengjie Zhao and Hinrich Schütze. Discrete and soft prompting for multilingual models. *arXiv preprint arXiv:2109.03630*, 2021.