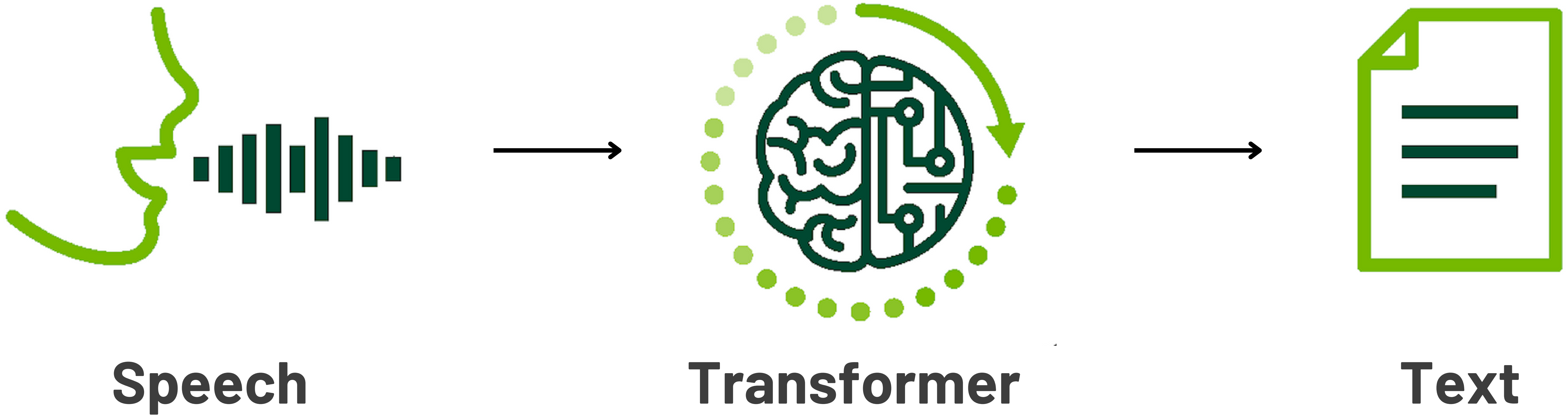
Automatic Speech Recognition

Fine-Tuning OpenAI Whisper on Italian Language



Barrasso Marco, Insaghi Edoardo

Università degli Studi di Trieste, Febbraio 2025

Reference Paper

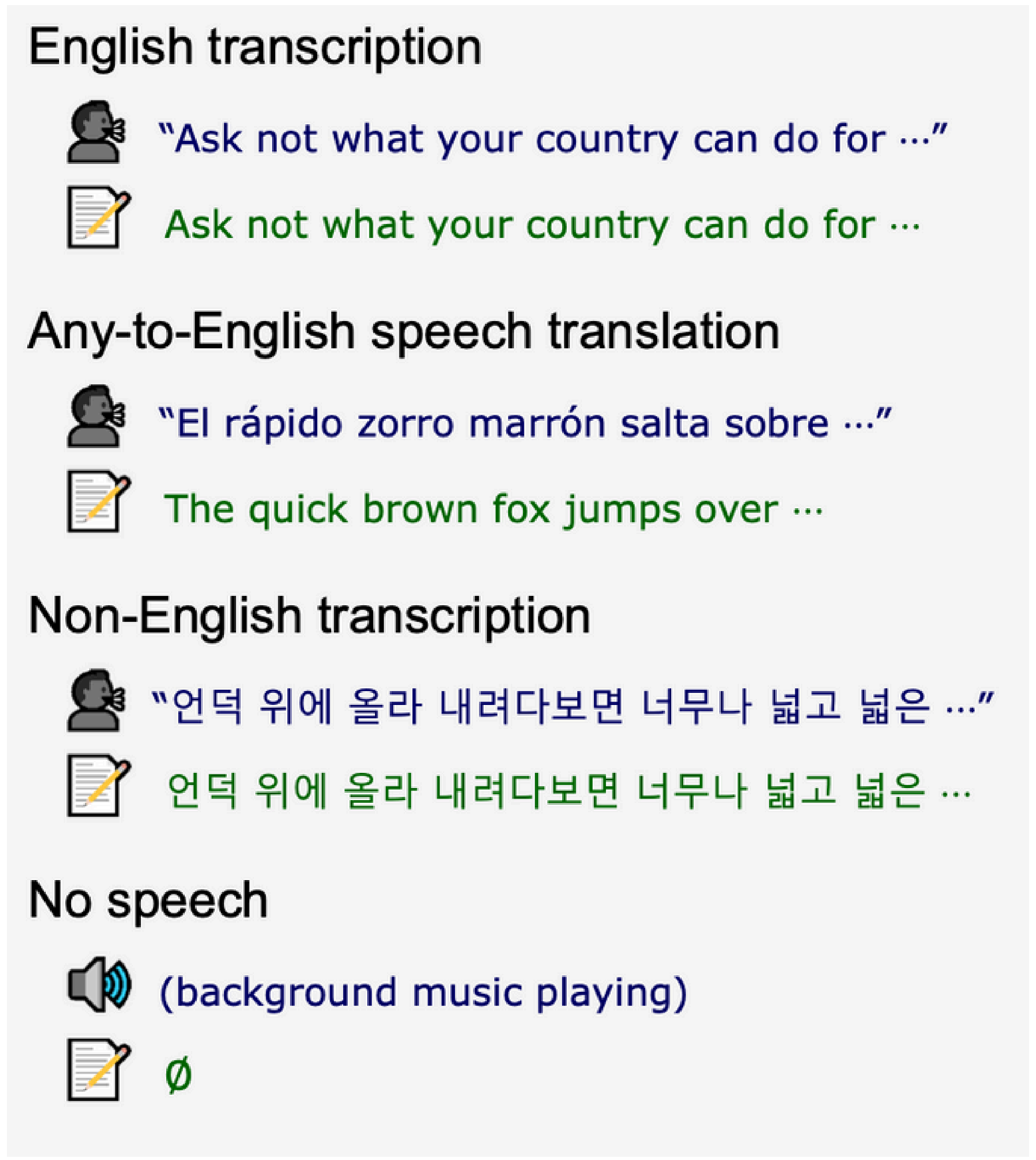
Robust Speech Recognition via Large-Scale Weak Supervision

Alec Radford [*1] Jong Wook Kim [*1] Tao Xu [1] Greg Brockman [1] Christine McLeavey [1] Ilya Sutskever [1]

Abstract

We study the capabilities of speech processing systems trained simply to predict large amounts of transcripts of audio on the internet. When scaled to 680,000 hours of multilingual and multitask supervision, the resulting models generalize well to standard benchmarks and are often competitive with prior fully supervised results but in a zero-shot transfer setting without the need for any fine-tuning. When compared to humans, the models approach their accuracy and robustness. We are releasing models and inference code to serve as a foundation for further work on robust speech processing.

Multitask training



English transcription

"Ask not what your country can do for ⋯"

Ask not what your country can do for ⋯

Any-to-English speech translation

"El rápido zorro marrón salta sobre ⋯"

The quick brown fox jumps over ⋯

Non-English transcription

"언덕 위에 올라 내려다보면 너무나 넓고 넓은 ⋯"

언덕 위에 올라 내려다보면 너무나 넓고 넓은 ⋯

No speech

(background music playing)

∅

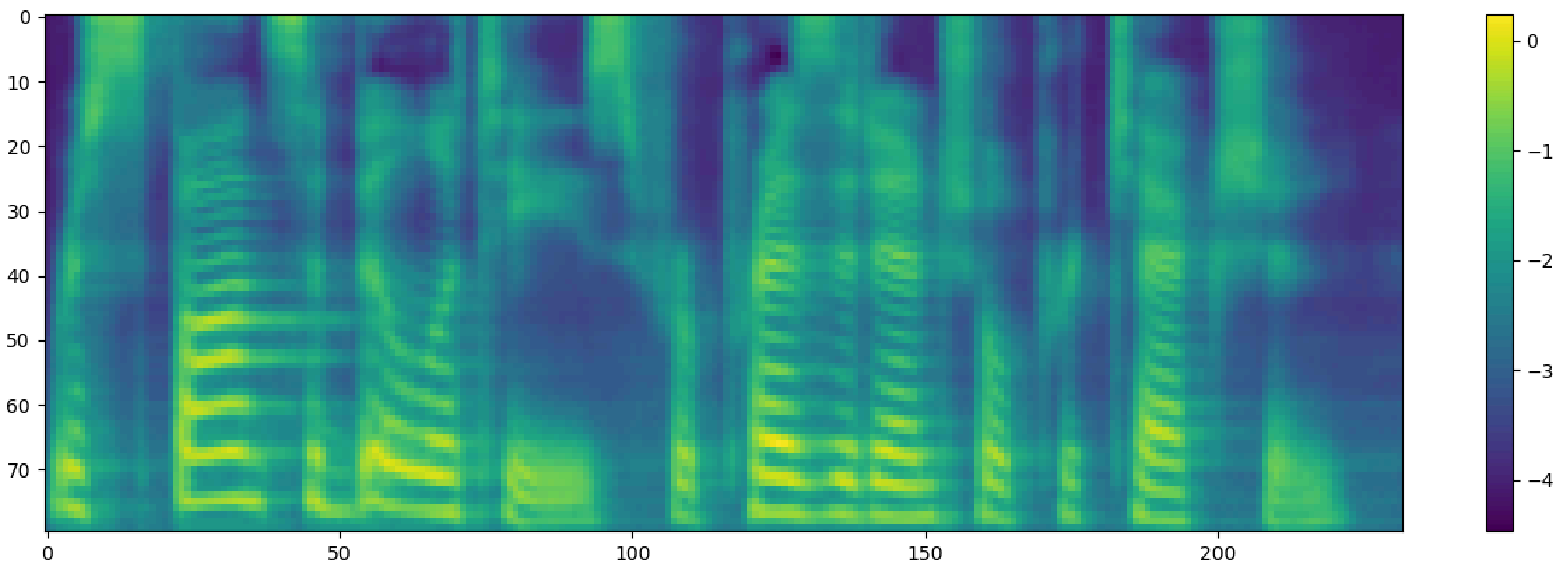
Working with Audio Data

Audio data is stored in computers as a **1 Dimensional waveform**

The waveform is a sequence of **amplitude values** sampled at a specific **sample rate** (16kHz and 44kHz are examples of standard sampling rate values)

The issue here is that the waveforms are often high-dimensional and complex for a neural network to extract information from

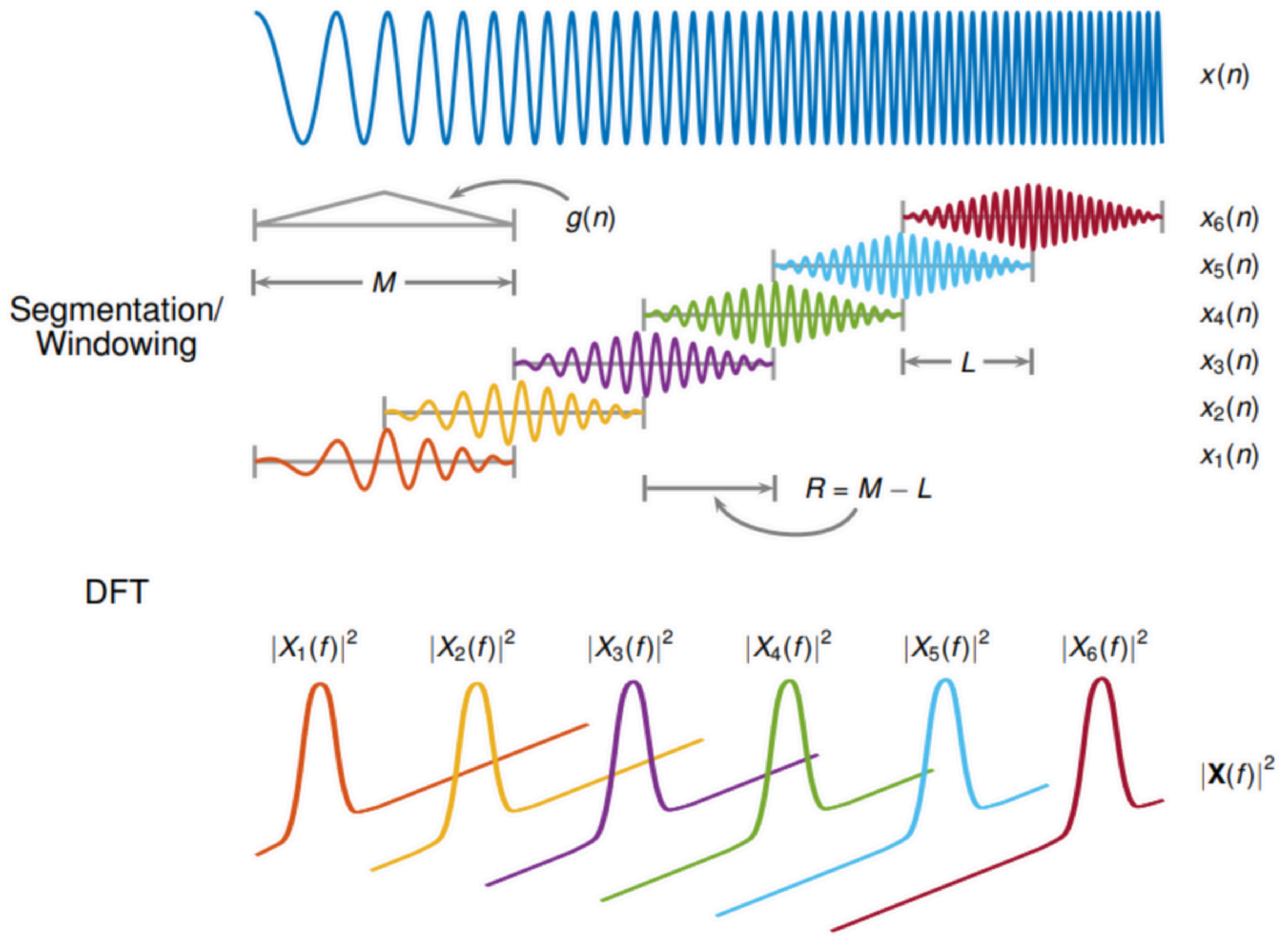
A standard solution consists of transforming the audio into a **Mel Spectrogram**. This allows to work with images that convey the audio information using the usual convolutional architectures



Example of a Mel Spectrogram

Building the Spectrogram

The **Fourier Transform** of a signal returns a list of complex coefficients telling us how much each frequency is present in the signal



The main idea is to compute the **Short Time Fourier Transform (STFT)**

$$X(m,k) = \sum_{n=0}^{N-1} x[n] \cdot w[n - mH] \cdot e^{-ikn}$$

(+ additional preprocessing and scaling)

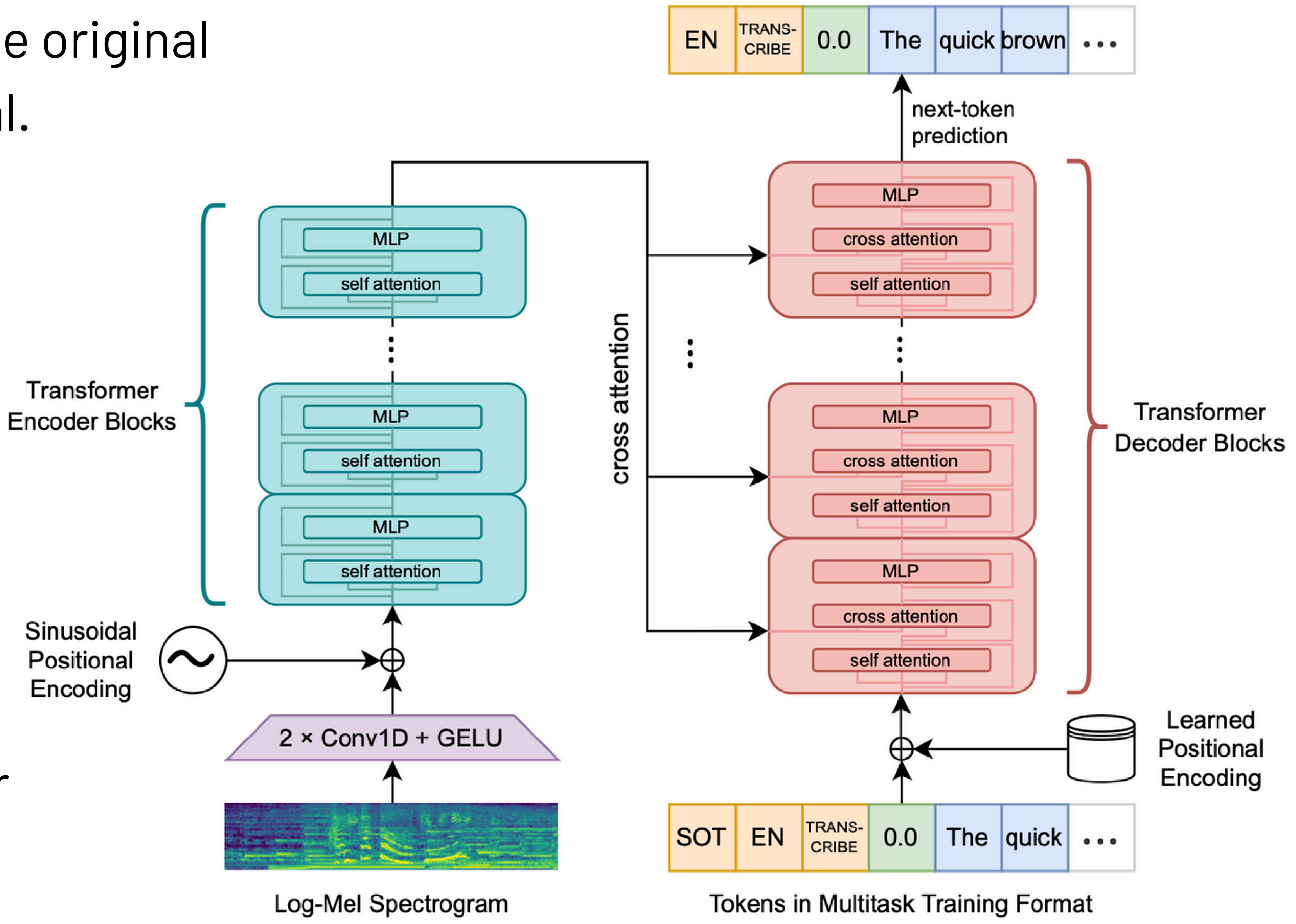
Which computes the Fourier Transform on overlapping windows of the original signal, telling how much influence each frequency has in different time windows

Model Architecture

The architecture is similar to the original **Transformer** from Vaswani et al.

With the addition of **Convolutional Layers** that process the information of the Mel Spectrograms before passing it to the **Encoder Layers** in the Transformer

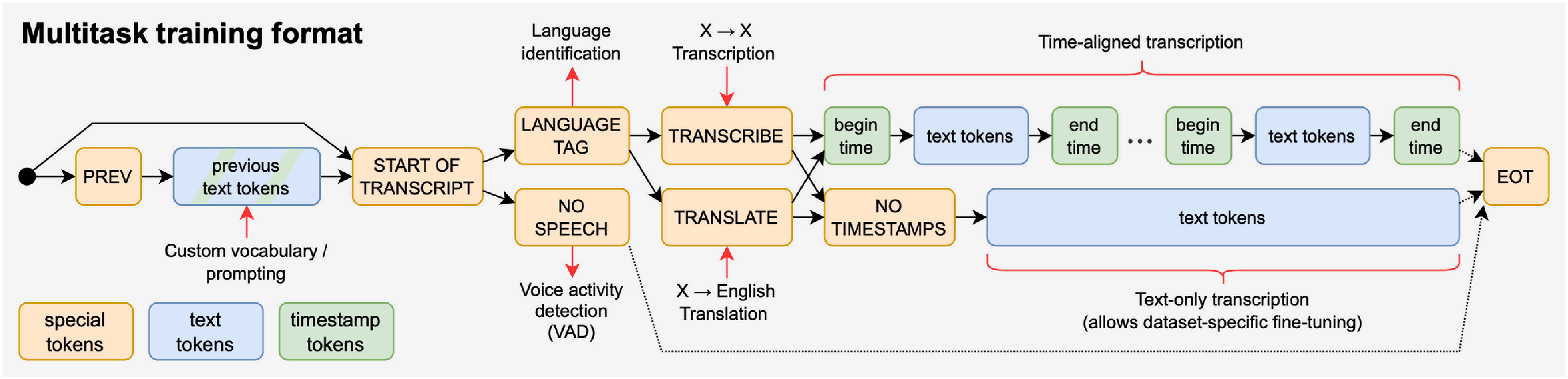
The authors also train a number of special tokens, needed for multitask training



Text Preprocessing

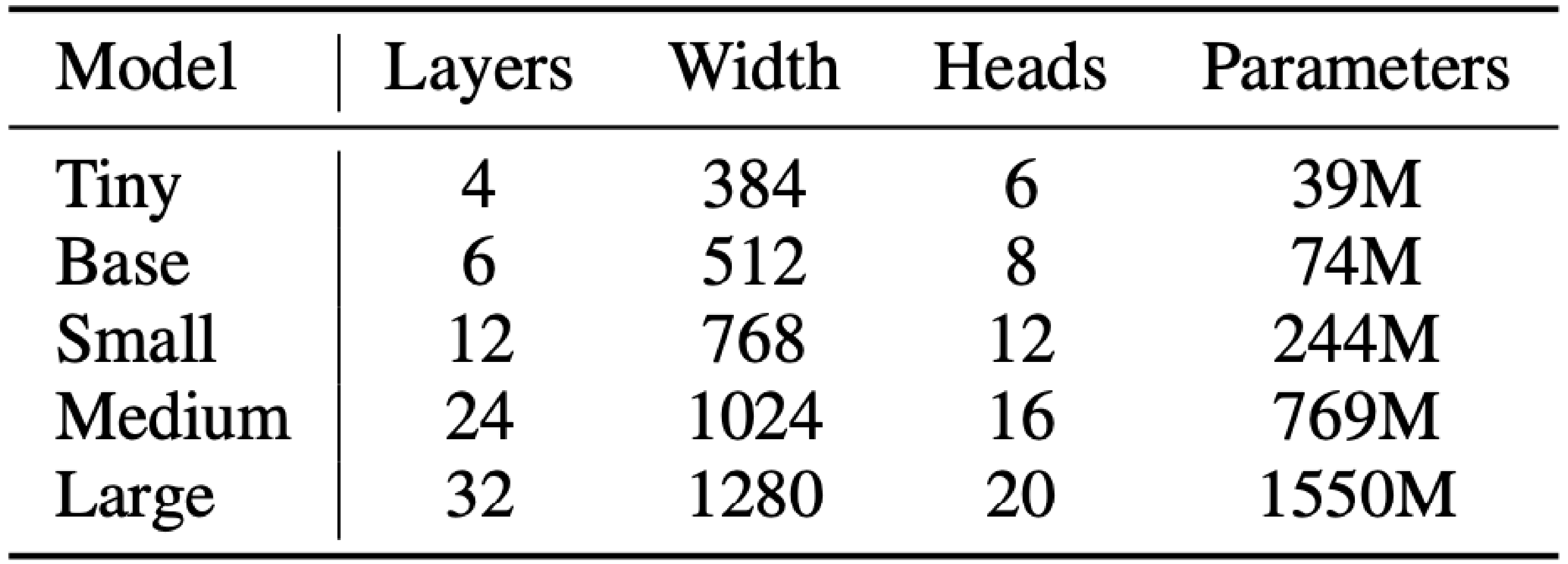
The authors use the standard GPT-2 tokeniser for the english only models, and retrain the vocabulary (with the same size) for the multilingual models to avoid excessive fragmentation on other languages

A number of special tokens is added to facilitate multitask training



Training Details

The paper from openAI describes the implementation of models of different sizes



Model	Layers	Width	Heads	Parameters
Tiny	4	384	6	39M
Base	6	512	8	74M
Small	12	768	12	244M
Medium	24	1024	16	769M
Large	32	1280	20	1550M

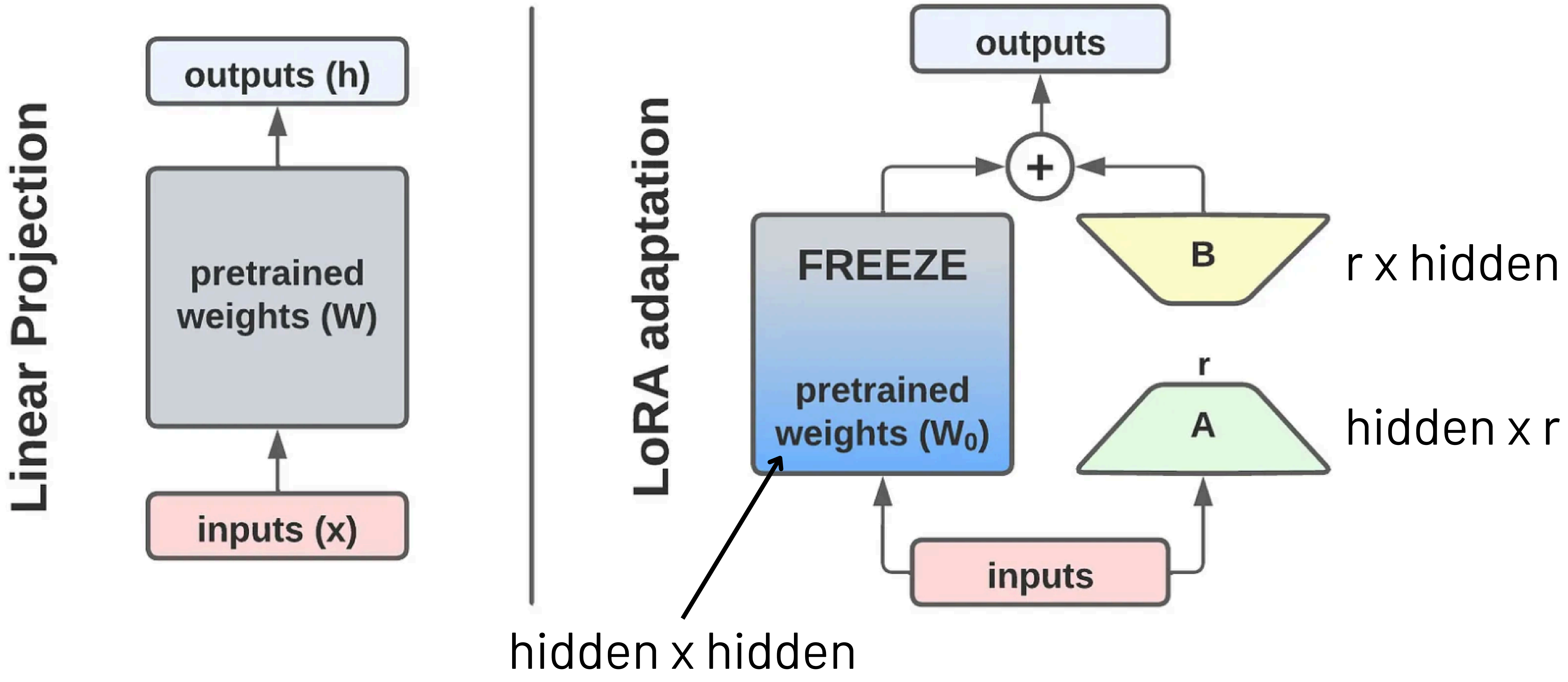
The model was trained on 680K hours of multilingual and multitask supervisioned data, with batches of 256 elements over 2^20 updates, which corresponds to less than 3 sweeps of the dataset

Generalisation comes from a dense sampling of the audio input space

Low Rank Adaptation

Training all the parameters of a large language model can be expensive

Low Rank Adaptation (LoRA) freezes the original parameters and trains additional matrices that will be added to the pretrained weights



Finetuning Experiments Setup

We decided to explore the effects of fine-tuning the model with two experiments:

1) We explore the difference between tuning all the weights of a small model versus LoRA (r = 32) on bigger models

Models	LoRA	no LoRA
Tiny	✓	✓
Base	✓	✗
Small	✓	✗

2) We investigate the effect of varying the LoRA hidden dimension on the performance of the “tiny” model, with different decoding algorithms.

Hidden dimensions of choice:
[16, 32, 64, 128]
Which account for ≈
[0.75%. 1.5%, 3%, 6%]
of the total number of parameters

Training Details

The train data consists of the Italian section of the **Multilingual LibriSpeech (MLS)** dataset.
It comprises more than 250 hours of speech coming from **LibriVox**

For our experiments we fix the number of training epochs to **2 sweeps** of the entire training dataset. The models are trained on a NVIDIA RTX 4090 gpu with the maximum batch size fitting in vRAM (24 GB)

We apply the LoRA parameters to all **Query and Value** weight matrices in every layer of the transformer, and we use a **linear decay learning rate scheduler** with warmup and AdamW optimiser.
The learning rate starts at 10^-3 for LoRA and 10^-4 for all weights fine-tuning

Evaluation Criteria

The choice for the evaluation metric is the commonly used **Word Error Rate (WER)**
Given a reference sentence and the hypothesis we obtain as the model's output,
the word error rate is defined as:

$$\mathrm{WER} = \frac{S + D + I}{N}$$

Where S, D and I stand for Substitutions, Deletions and Insertions,
while N is the total number of words in the ground truth.

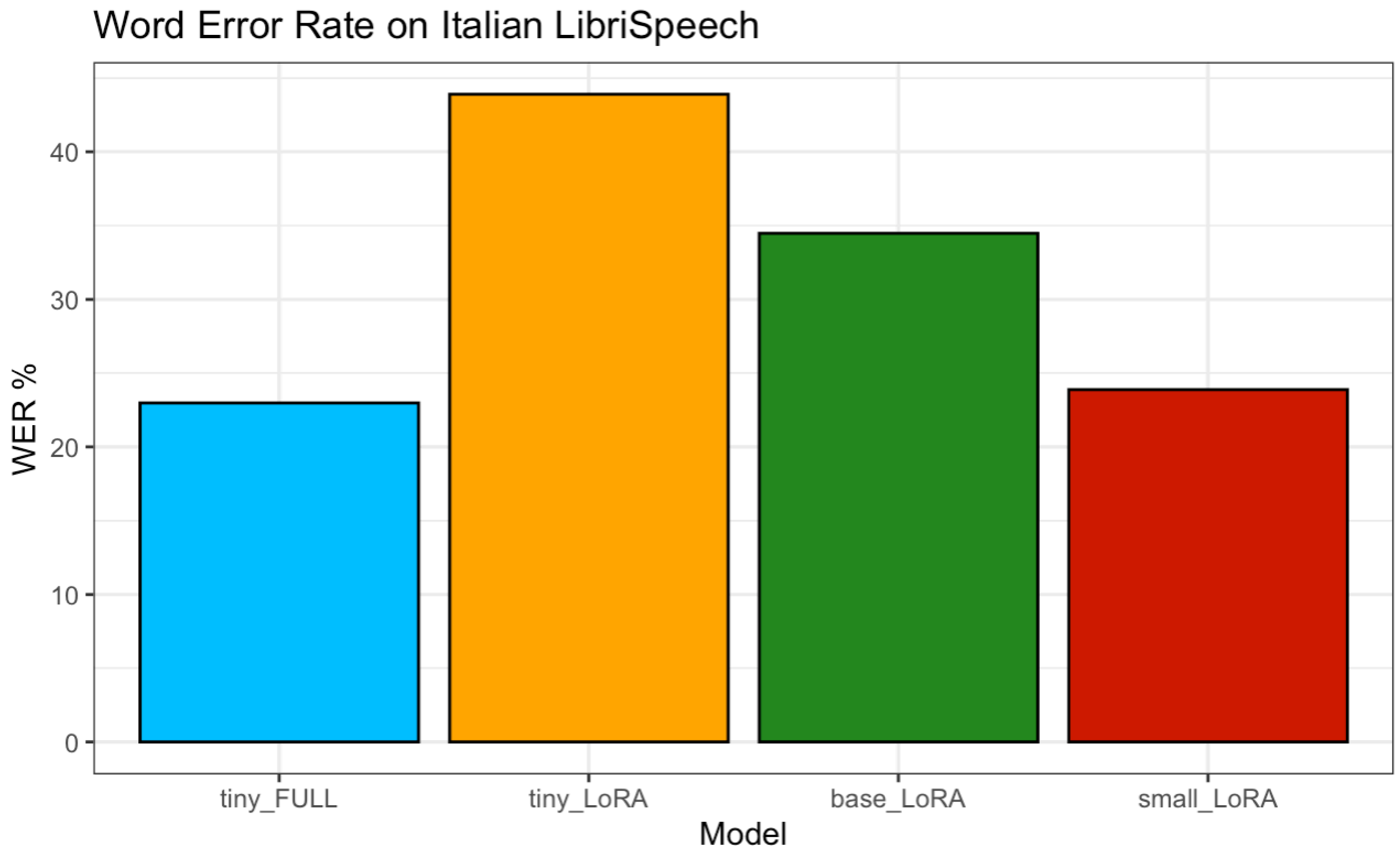
We use the WER as a metric to evaluate how well the model performs on Italian
after the tuning. We also use it to measure the loss in performance on English data.

Results

Bigger models lead to a **higher** performance

The performance of the fully fine-tuned tiny model is **comparable** with the LoRA fine-tuning of the small model

We did not observe significant **speedup** during training with LoRA, possibly because of the relatively limited size of the models

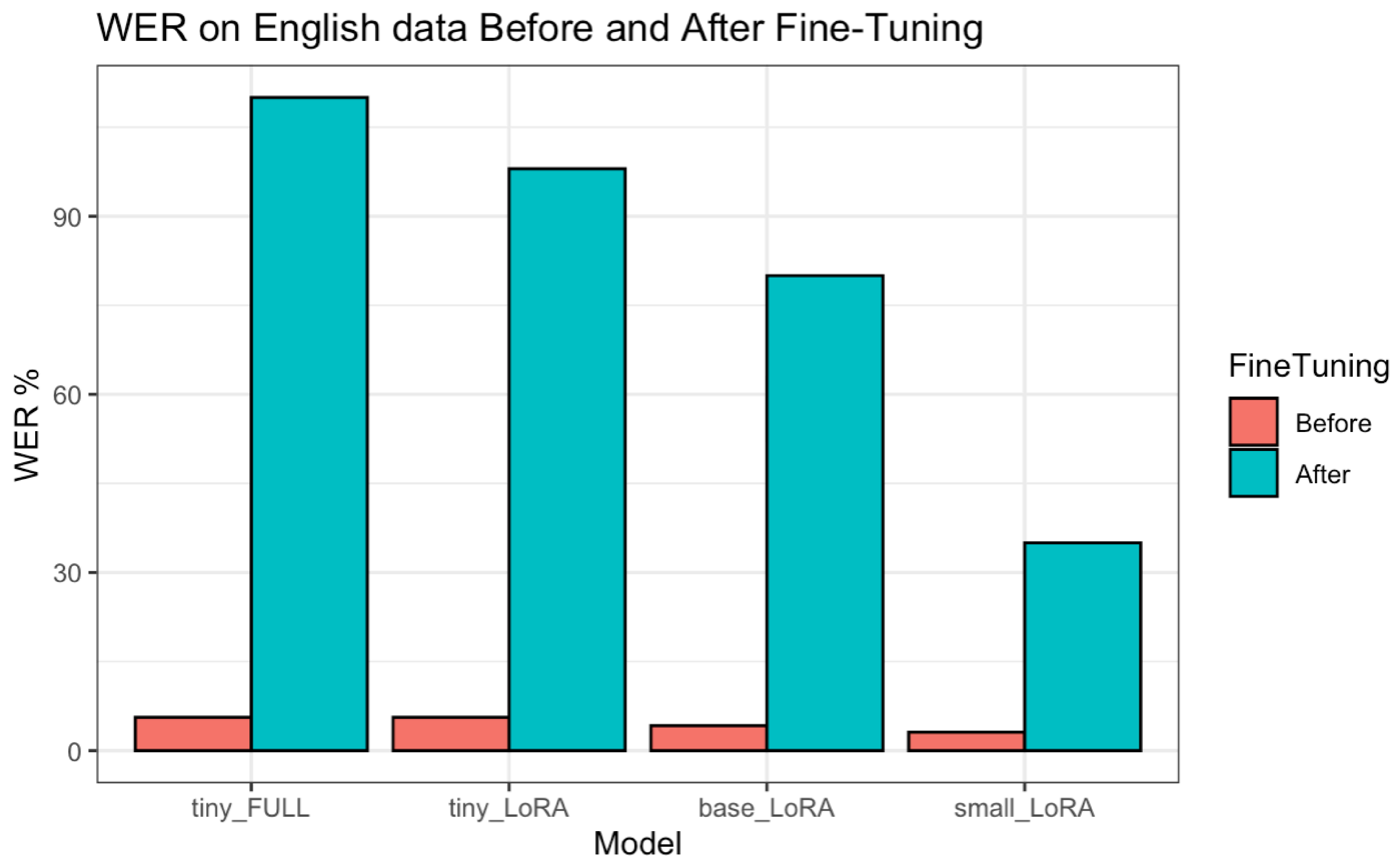


Results

The models **tend to forget** the English language after fine-tuning

The fully fine-tuned model seems to perform **worse** than its LoRA counterpart

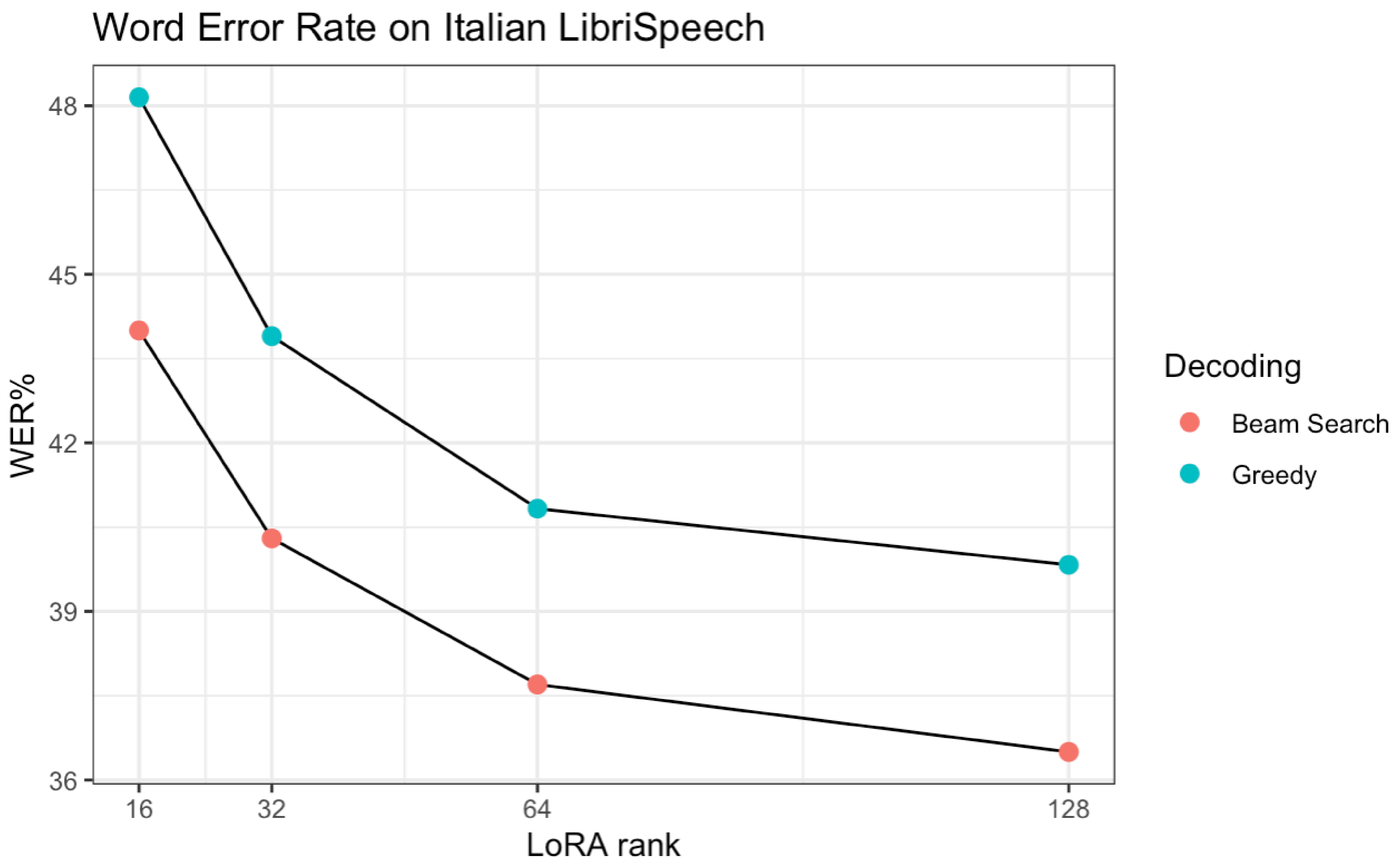
The **bigger** the model is, the **better** it preserves its memory about past information



Results

Beam search **outperforms** greedy decoding at the expense of a higher **computational cost**

A higher LoRA rank leads to better results with apparently diminishing returns



Thanks for the $\mathbf{softmax}(\frac{\mathbf{QK^T}}{\sqrt{\mathbf{d_k}}})\mathbf{V}$