A Supplemental materials

A.1 Model & hyperparameters

All LM models are trained with a batch size of 64, where each sample is bounded for 25 seconds and 704 tokens. The models are trained for 400k steps ($\sim$1.2 epochs), using an inverse-sqrt scheduler, 100 warmup steps and wADAM as the optimization algorithm. We also tune the learning rate per scenario, i.e: using/not-using pretrained LM, we end up with a maximal learning rate of 4e-4/8e-5 and final learning rate of 8e-5/2.5e-5, respectively. As for the LLaMA-7B model, we use the same configuration except the following: cosine learning rate schedule, 500 warmup steps, a maximum learning rate of 1e-4, a final rate of 1e-5, batch size of 1024 over 32 GPUs for 75k steps ($\sim$4 epochs).

The HuBERT speech tokenizer, which is not part of the textless-lib Kharitonov et al. [2022] (i.e., the 25Hz frequency model), is trained for 3 iterations with the default 50Hz features rate. For the 4-th iteration, we add an additional convolutional layer at the CNN Encoder with the strides 2/3/4, resulting in features of 25Hz/16.6Hz/12.5Hz, respectively. Our early ablations show that 25Hz features with 500 tokens give the best results in terms of language modeling, we thus train our models on these new tokens and compare them with the rest of the tokens.

A.2 Speech resynthesis results

Resynthesis can be considered as an upper bound for our language modeling setup. It does not involve SpeechLMs, and measures our ability to fully recover the speech content after tokenization [Polyak et al., 2021]. As we additionally evaluate several speech tokenizers, we provide resynthesis metrics in the form of Word Error Rate (WER). We use Whisper [Radford et al., 2022] "small" as our ASR model.

In Table 5, we evaluate the effect of the tokenizer on the resynthesis performance, and can better evaluate the impact of the tokenization process on the generated audio. As can be seen, all tokenizers incur a loss in WER. Using 500 tokens at 25Hz provides the best performance.

Table 5: Speech Resynthesis. Results are reported for different number of tokens and downsampling factors (Frequency).

# TOKENS	FREQUENCY	WER↓
100	50Hz	0.23
200	50Hz	0.18
500	50Hz	0.17
500	25Hz	**0.16**

A.3 Model and data scaling results

The full set of results, i.e., PPL, sWUGGY and sBLIMP from Section 4 for model and dataset scaling can be seen on Table 6. The equivalent of Fig. 2a using 200 tokens at 50Hz tokenizer can be found in Fig. 5.

A.4 The effect of LM architecture

To further validate our findings holds for other LM architectures other than OPT, in Table 7, we provide results for the BLOOM [Scao et al., 2022] and Pythia [Biderman et al., 2023].

As before, we observe similar patterns in terms of using a pretrained text LM. SpeechLMs initialize from text reach better performance across all metrics.

A.5 The effect of different modality pretraining

Although having completely different granularity, results suggest training SpeechLMs with model initialization from a text based LMs brings a consistent performance improvement. As a result, a natural question would be *do speech and text tokens have special connection or LMs are just general next token prediction mechanisms?*

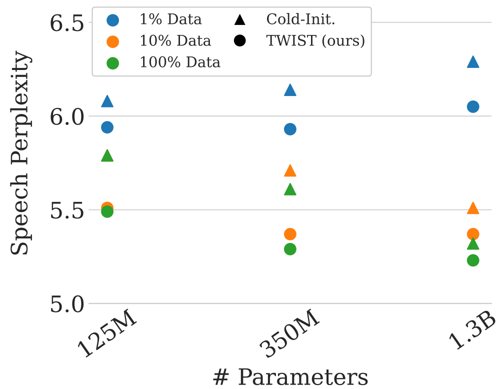

Figure 5: PPL as a function of training data set for the 200 tokens at 50Hz tokenizer.

Table 6: Model and Data Scaling. Results are reported for different models on various size using different magnitude of data, with and without TWIST. We report PPL/ sWUGGY / sBLIMP.

TWIST	# PARAM.	# TOKENS	FREQ.	1% OF DATA	10% OF DATA	100% OF DATA
✗	125M	200	50Hz	6.08 / 65.94 / 54.17	5.79 / 67.47 / 54.52	5.79 / 67.62 / 54.58
✓				**5.94 / 68.72 / 54.62**	**5.51 / 69.91 / 56.42**	**5.49 / 70.20 / 56.02**
✗	350M	200	50Hz	6.14 / 65.61 / 53.90	5.71 / 68.17 / 55.14	5.61 / 68.68 / 55.55
✓				5.93 / **67.77 / 54.79**	**5.37 / 71.79 / 57.71**	**5.29 / 71.83 / 57.89**
✗	1.3B	200	50Hz	6.29 / 64.52 / 54.00	5.51 / 70.89 / 56.71	5.32 / 71.86 / 57.02
✓				**6.05 / 67.43 / 54.32**	**5.37 / 71.97 / 57.81**	**5.23 / 72.51 / 58.04**
✗	125M	500	25Hz	7.22 / 78.77 / 56.58	6.82 / 79.35 / 57.13	6.81 / 79.19 / 57.40
✓				**7.06 / 79.97 / 57.52**	**6.52 / 81.23 / 58.83**	**6.50 / 81.44 / 58.78**
✗	350M	500	25Hz	7.37 / 77.96 / 56.29	6.79 / 78.97 / 57.52	6.65 / 80.38 / 58.00
✓				**7.26 / 79.92 / 57.06**	**6.41 / 82.41 / 59.60**	**6.26 / 82.40 / 59.31**
✗	1.3B	500	25Hz	7.49 / 77.10 / 55.82	6.40 / 81.59 / 58.98	6.20 / 82.69 / 59.55
✓				**7.19 / 79.52 / 56.87**	**6.19 / 83.07 / 59.94**	**6.04 / 82.66 / 60.46**

Table 7: LM Model Architecture. Results are reported for both Bloom and Pythia model architectures, with and without TWIST. We report PPL, sWUGGY and sBLIMP.

TWIST	ARCH.	# TOKENS	FREQ.	PPL↓	SWUGGY↑	SBLIMP↑
✗	Bloom	200	50Hz	5.63	68.51	55.26
✓				**5.21**	**71.51**	**57.90**
✗	Bloom	500	25Hz	6.45	81.01	58.95
✓				**6.06**	**82.92**	**60.52**
✗	Pythia	200	50Hz	5.62	69.65	55.42
✓				**5.23**	**71.40**	**58.02**
✗	Pythia	500	25Hz	6.45	81.07	59.00
✓				**6.12**	**82.45**	**60.06**

To evaluate such a hypothesis, we consider a language model trained on a different modality. Specifically, we train ImageGPT [Chen et al., 2020] ("medium" size) models, one from scratch and another one pretrained using next pixel prediction using a transformer language model. For the pretrained model we use the official pre-trained model.[8] Table 8 summarizes the results.

Interestingly, ImageGPT pre-trained models perform much worse than models pretrained on text. For a reference, models trained from scratch achieve comparable performance to previously reported models.

[8] https://huggingface.co/docs/transformers/model_doc/imagegpt

Table 8: Results for the ImageGPT model (image pretraining), with and without TWIST. We report PPL, sWUGGY and sBLIMP. Unlike textual pretraining, image pretraining not only does not benefit SpeechLMs, but substantially hurts their performance.

TWIST	# TOKENS	FREQ.	PPL↓	sWUGGY↑	sBLIMP↑
✗	200	50Hz	**5.22**	**71.47**	**58.16**
✓			8.21	55.02	53.34
✗	500	25Hz	**6.20**	**82.38**	**59.55**
✓			7.85	74.36	54.55