Slamming: Training a Speech Language Model on One GPU in a Day

Gallil Maimon* **Avishai Elmakies*** **Yossi Adi**
*Equal Contribution
The Hebrew University of Jerusalem
`gallil.maimon@mail.huji.ac.il`

Abstract

We introduce *Slam*, a recipe for training high-quality Speech Language Models (SLMs) on a single academic GPU in 24 hours. We do so through empirical analysis of model initialisation and architecture, synthetic training data, preference optimisation with synthetic data and tweaking all other components. We empirically demonstrate that this training recipe also scales well with more compute getting results on par with leading SLMs in a fraction of the compute cost. We hope these insights will make SLM training and research more accessible. In the context of SLM scaling laws, our results far outperform predicted compute optimal performance, giving an optimistic view to SLM feasibility. See code, data, models, samples - https://pages.cs.huji.ac.il/adiyoss-lab/slamming.

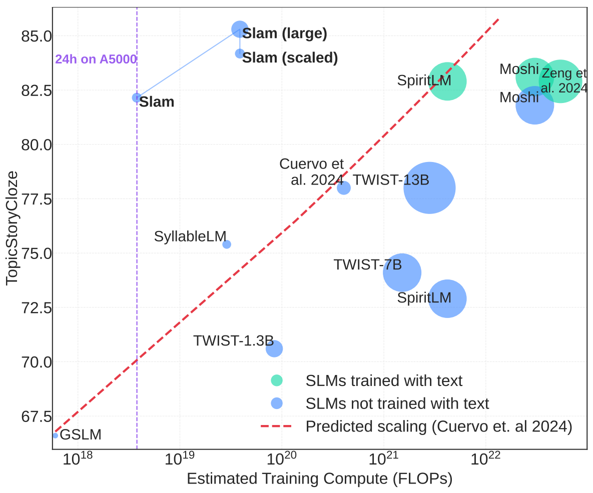

Figure 1: Comparing Topic-StoryCloze performance of different SLMs as a function of training compute. Model size is indicated by the size of the circle.

1 Introduction

Speech Language Models (SLMs) have gained significant interest from researchers (Peng et al., 2024a; Cui et al., 2024; Ji et al., 2024; Latif et al., 2023), demonstrating remarkable performance in traditional speech tasks (Wang et al., 2023; Elmakies et al., 2025), diverse generative applications (Yang et al., 2023, 2024b), and reasoning over speech and audio signals (Tang et al., 2024; Chu et al., 2023).

SLMs can generally be classified into two main categories: (i) generative speech Language Models (LMs) (which can also incorporate text) and (ii) speech-aware LMs. The first category follows a similar pre-training approach to text-based Large Language Models (LLMs), directly maximising the likelihood of speech considering both input and output, typically by representing audio as a sequence of discrete tokens. The second category consists of pre-trained text LMs adapted to process speech inputs. In this work, we focus on the first.

Training high-quality SLMs can be highly resource intensive (Hassid et al., 2024; Cuervo and Marxer, 2024; Zeng et al., 2024; Nguyen et al., 2025; Défossez et al., 2024). For example, Nguyen et al. (2025) trained their SLM on approximately $570k$ hours of speech data, while Défossez et al. (2024) utilised around $7M$ hours. Additionally, Cuervo and Marxer (2024) proposed SLM scaling laws, suggesting that training high-quality SLMs requires $\sim 3X$ more data compared to text-based counterparts. These computational demands restrict the required fundamental research aimed at enhancing SLMs, such as advancements in speech tokenisation, efficient acoustic modelling, etc.

In the Natural Language Processing (NLP) community, numerous studies have investigated efficient model training techniques, including masked language models such as Cramming (Geiping and Goldstein, 2023) and ModernBERT (Warner et al., 2024), along with next-token prediction LLMs such as MobileLLM (Liu et al., 2024b). These methods include implementation efficiencies, architectural improvements, data selection strategies, and enhancements to the overall training pipeline.

Inspired by Cramming (Geiping and Goldstein, 2023) in text, we investigate compute-limited SLM training, which we term *Slamming*. We pose the question: *Is it possible to train high-quality SLMs using a single GPU within 24 hours?* For that, we conduct an extensive empirical analysis exploring how different training components influence performance. From this, we derive a training recipe that maximises model performance within a fixed compute budget. Specifically, we investigate the impact of model initialisation and architecture, various optimisers and learning rate schedulers, data selection strategies - including the role of synthetic data, text-interleaving and preference optimisation.

We believe that developing these training strategies and proving their feasibility will empower the speech and audio research community to advance SLMs beyond the scope of large, well-funded academic and industrial labs. Figure 1 illustrates the performance of various SLMs relative to their training compute budget, with circle sizes representing the size of the models. Furthermore, we compare our results with the scaling performance predicted from Cuervo and Marxer (2024). Although the authors present a somewhat pessimistic view of the computational resources needed to train high-quality SLMs, we empirically show that reality is more promising, demonstrating that it is possible to significantly exceed the predicted performance per unit of compute. We encourage the community to refine and expand scaling laws specifically tailored for SLM training across various settings.

Our Main Contributions are:

1. We introduce *Slam*, a training recipe for efficiently training high-quality SLMs using a single $A5000$ GPU within 24 hours.

2. We carry out extensive experiments exploring model initialisation and architecture, optimisation, data collection and generation, and training objectives (i.e., preference optimisation and text-speech interleaving), providing insights into the impact of each component on model performance.

3. Building on these insights, we scale the compute budget to two $A100$ GPUs for 48 hours and demonstrate that our model achieves performance on par with state-of-the-art models that require substantially more compute.

We open-source all code, models, training recipes, and synthetic datasets.

2 Related Work

Efficient Training. Enhancing the efficiency of neural network training has been extensively studied (Shen et al., 2023). Hajimolahoseini et al. (2023); Wang et al. (2024) examined the impact of data selection on Large Language Model (LLM) training and introduced efficient data selection methods. Muhamed et al. (2024) proposed using structured sparse gradients to enhance compute efficiency in LLM training, while Rawat et al. (2024) explored the potential of leveraging smaller language models to improve the training efficiency of larger LLMs. Lv et al. (2024) investigated the use of low-dimensional projections for attention parameters to enhance training efficiency. Meanwhile, Neiterman and Ben-Artzi (2024) proposed applying LayerDrop as a technique to optimise neural network training.

More closely related to our work, Li et al. (2023) propose a training strategy for developing LLMs within a $100k\$$ budget. Warner et al. (2024) introduce ModernBERT, an efficient training pipeline for optimising BERT models, while Izsak et al. (2021) outline a method for training a BERT model in 24 hours using 8 GPUs. The most relevant work to ours is Cramming (Geiping and Goldstein, 2023), where the authors conduct an in-depth analysis of masked LM training on a single GPU in one day.

While these studies offer valuable insights, they primarily focus on training text models, such as LLMs and masked LMs. In the speech domain, similar research has been conducted on self-supervised representation models (Liu et al., 2024a), but not on SLMs. In this work, we address this gap by focusing on efficient SLM training.

Generative Speech Language Models were explored under various setups (Lakhotia et al., 2021; Kharitonov et al., 2021). Lakhotia et al. (2021) were the first to show how raw, uncurated speech data can be leveraged into building a Generative Speech Language Model (GSLM). Next, Borsos et al. (2023) proposed a cascade version using both coarse and fine speech tokens. Such a modelling framework opened up a new and promising research direction for processing and modelling spoken data, such as speech resynthesis (Polyak et al., 2021), speaking style conversion (Kreuk et al., 2021; Maimon and Adi, 2023), dialogue modelling (Nguyen et al., 2022), speech-to-speech translation (Popuri et al., 2022; Peng et al., 2024b), etc. Nachmani et al. (2024) proposed augmenting a text Language Model (LM) with continuous speech data to improve spoken question-answering tasks. Recently, Park et al. (2024) proposed SLM based on state-space models (Gu et al., 2021) to further push long context-efficient modelling, while Lin et al. (2024) proposed to fine-tune SLMs using direct preference optimisation (Rafailov et al., 2024) obtained from text LLM rankings.

Similar to text LLMs, training SLMs often demands large-scale datasets. For instance, Moshi (Défossez et al., 2024) was trained on 7 million hours of speech data, SpiritLM (Nguyen et al., 2025) utilized $560k$ hours, and TWIST (Hassid et al., 2024) was trained on approximately $150k$. Recently, Cuervo and Marxer (2024) introduced the first scaling laws for SLMs, suggesting that achieving comparable performance to text LMs requires three times more tokens. In this work, we focus on reducing the computational demands while maintaining performance comparable to leading SLMs.

3 Setup

In this study, we explore decoder-only generative SLMs, which aim at maximising the likelihood of speech samples represented as discrete tokens. We examine both purely speech-based SLMs trained on speech tokens and joint speech-text SLMs using interleaving strategies (Nguyen et al., 2025). Similarly to Hassid et al. (2024); Lakhotia et al. (2021), we obtain speech tokens by quantising continuous latent representations of a self-supervised speech representation model using the k-means algorithm, often known as *semantic tokens*. Specifically, we utilise a multilingual HuBERT (Hsu et al., 2021) model running at 25 Hz, as employed in Hassid et al. (2024). We then train SLMs by minimising the negative log-likelihood of the input segments.

Unless mentioned otherwise, all SLMs are trained using **a single** $A5000$ **GPU** ($24GB$ **VRAM)** along with 16 CPU cores for 24 hours. We deliberately focus on this constrained compute budget, assuming that most academic labs can access similar resources, thereby ensuring the accessibility of our research. The training data is pre-processed, i.e. extracting HuBERT units and dividing data into chunks, and stored prior to model training. As a result, this pre-processing time is excluded from the compute budget. This approach, aligned with Geiping and Goldstein (2023), is practical since many research experiments utilise the same pre-processed data. We additionally do not count the time for running validation and visualisations as they are not used as part of the optimisation pipeline and only used for demonstration purposes.

Evaluation Metrics. We assess all SLMs using five distinct evaluation metrics. The first three are based on likelihood evaluation, while the fourth and fifth are generative metrics. For likelihood based modelling we consider sBLIMP (Dunbar et al., 2021), *Spoken Story-Cloze* (sSC)), and *Topic Story-Cloze* (tSC) (Hassid et al., 2024). For modelling-likelihood metrics, we evaluate the likelihood assigned by the SLMs to pairs of speech utterances, consisting of a positive example and a distractor. We calculate the percent of pairs in which the SLM assigns higher likelihood to the positive sample. sBLIMP focuses on grammatical abilities thus the negative is ungrammatical version of the positive. sSC and tSC focus on semantic modelling abilities. In sSC, the distractor suffix is taken from the original textual StoryCloze dataset (Mostafazadeh et al., 2016), allowing to assess fine-grained semantic speech understanding. In tSC, however, the distractor suffix is drawn from a different topic, enabling us to evaluate the model's ability to understand the overall semantic concept.

To assess the generative abilities of SLMs, we compute *generative perplexity* (GenPPL). Following the approach of Lakhotia et al. (2021); Hassid et al. (2024), we provide the SLM with a short speech prompt and generate speech tokens continuation. We use unit-vocoder with duration prediction to convert the tokens into speech (Polyak et al., 2021; Hassid et al., 2024). The generated speech is then transcribed, and its Perplexity (PPL) is evaluated using a pre-trained text LLM. To minimise the impact of token repetition on PPL measurements, we ground the generated text using diversity metrics derived from the auto-BLEU score (Lakhotia et al., 2021). Similarly to Lin et al. (2024) we use bigram auto-BLEU. In other words, we ensure that all models achieve similar auto-BLEU scores, allowing for a fair comparison of PPL. Specifically, we transcribe speech segments using Whisper-large-v3-turbo model (Radford et al., 2023) and measure PPL using Llama-3.2-1B model (LLama, 2024). We calculate GenPPL on correct samples from the Spoken Story-Cloze dataset.

Finally, for our final models, we also compute

GPTScore. Given a speech prompt and a generated continuation, we transcribe both and use GPT-4o to judge the quality of the continuation given the prompt, on a scale of 1 to 5. We follow the same setup and prompt as Lin et al. (2024) for the metric. We use this metric as the final form of evaluation, as it is the most costly to run.

Software Efficiency. To maximise performance within 24 hours of model training, we leverage multiple efficient implementations. Through extensive performance testing, we found that using bfloat16 (Kalamkar et al., 2019) alongside FlashAttention2 (Dao, 2023) and data packing provided the most efficient compute performance in our setup. We also experimented with model compilation using `torch.compile` (Ansel et al., 2024), but it lacked native compatibility with FlashAttention2 at the time of our study, and its performance without FlashAttention2 was subpar. Future work could investigate this further with more efficient attention implementations (Shah et al., 2024; Li et al., 2024).

To enable rapid and scalable experimentation, we developed a specialised library for SLM training that supports various model architectures, training objectives, and evaluation metrics. It accommodates TWIST-style training, text-speech interleaving, preference optimisation, etc. We open-source this package along with all model weights and training recipes, aiming to empower the community to further explore SLMs.

4 Investigations

With this setup, we systematically analyse and ablate each component of the training pipeline, ultimately refining an optimised cook-book for training SLMs. We specifically examine the influence of model family, initialisation, size, and architectural choices (e.g., dropout, positional embedding, etc.). We analyse optimisation parameters and data characteristics. Lastly, we explore alternative training objectives beyond standard next-token prediction, including speech-text interleaving and direct preference optimisation using synthetic data.

4.1 Model & Optimisation

Hyper-parameters. Unless specified otherwise, we use a context length of 512 tokens and an effective batch size of 256, employing gradient accumulation when necessary, as preliminary results indicated this configuration yields the best overall performance. We set the peak learning rate to $1e{-}3$

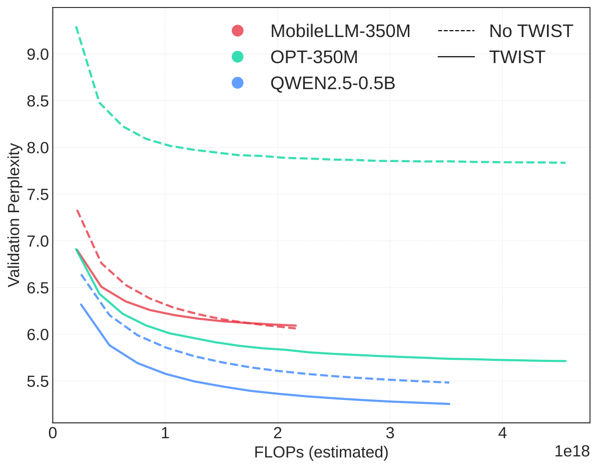

Figure 2: Comparing validation PPL of different models of similar parameter count, with and without TWIST initialisation.

to enhance training speed and use a warmup period of 1% of the total training steps, as this proved more effective than the fixed 100-step warmup used in the original TWIST. To improve training stability, particularly with large learning rates, we apply gradient normalisation with a norm of 0.5 at no additional cost, following Geiping and Goldstein (2023). Unless modified later in our investigation, we use an inverse-square root scheduler and the AdamW optimiser (Loshchilov, 2017). Likewise, this sub-section uses the common Libri-Speech And Libri-Light datasets for training, until further investigated in Section 4.2.

Initialisation. Hassid et al. (2024) empirically demonstrated that initialising SLMs with pre-trained text LMs can enhance convergence speed and improve model performance. We examine the effect of this initialisation within our setup across different model types. To do so, we train multiple models, both with and without TWIST initialisation, while staying within our compute budget. As shown in Figure 2, TWIST initialisation benefits all evaluated models at the beginning of training, though its overall impact by the end varies. Notice, the x-axis in Figure 2 represents theoretical FLOPs, calculated as $6 * N_{\text{params}} * D_{\text{tokens}}$ following Hoffmann et al. (2022). However, due to variations in model architecture and implementation, practical efficiency differs, leading to varying amounts of compute processed within 24 hours.

Results suggest that benefits of TWIST initialisation can be substantial, especially for top-performing models like Qwen2.5. As a result, we prioritise investigations based on existing pre-

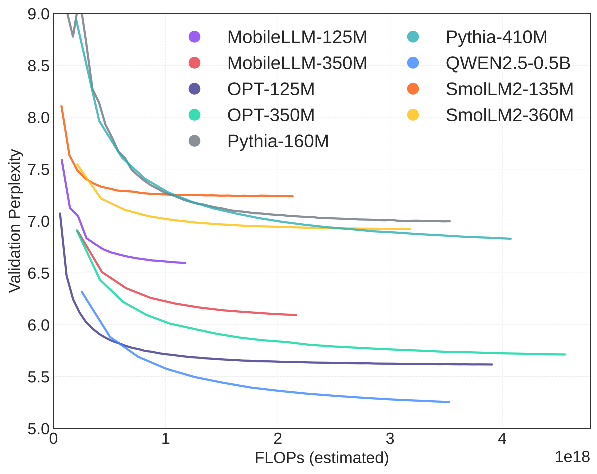

Figure 3: Comparing PPL of different models under TWIST initialisation.

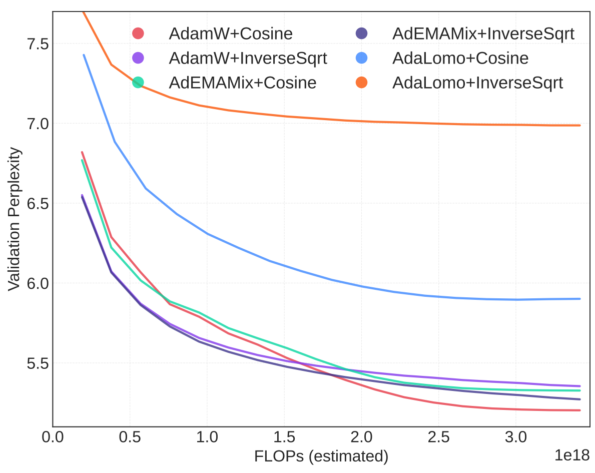

Figure 4: Comparing validation PPL of our best model with different optimisers and schedulers.

trained text LMs. Interestingly, the results in Figure 2 demonstrate that Qwen2.5 outperforms other models even without TWIST initialisation, perhaps suggesting that their architectural design choices or size might also provide some benefit.

Optimal Model Size & Family. Cuervo and Marxer (2024) conducted a scaling analysis on GSLM-style SLMs, estimating the optimal model size and token count for a compute-efficient model. However, using a text LM initialisation might impact these findings. As we observe, TWIST initialisation greatly impact model performance, suggesting that prioritising larger models may be more effective than simply increasing the dataset size. Additionally, various model families gain different advantages from TWIST initialisation; for example, Qwen2.5 models show significantly better performance compared to OPT models. In Figure 3, we compare the results under the pre-defined compute budget within model families[1]. We note that the best model sizes for MobileLLM (Liu et al., 2024b), SmolLM2 (Allal et al., 2025) and Pythia (Biderman et al., 2023) are $\sim 300M$ parameters, while for OPT the best is 125M. According to Cuervo and Marxer (2024), the estimated optimal model size is approximately 66M parameters. However, the best-performing model, Qwen2.5, is significantly larger. Since there are no smaller models in this family, it is difficult to determine whether this deviation is due to the quality of the initialisation or other factors. Moving forward, we proceed with both OPT-125M and Qwen2.5-0.5B.

Dropout. The original OPT models includes dropout to mitigate overfitting. Although dropout is beneficial for regularisation, it effectively decreases the number of gradient updates per parameter without shortening the update-step wall time. Hence, reduces the number of parameter updates per second. Following Geiping and Goldstein (2023), we experiment with removing dropout and observed improved performance in our setup.

Positional Encoding. Transformers rely on positional encoding to capture the order of input tokens. Many modern LMs, including the Qwen models, use Rotary Position Embedding (Su et al., 2024). This method uses a hyperparameter, θ, to control the trade-off between granularity and the ability to handle long contexts. θ is often tuned to accommodate longer context lengths (Yang et al., 2024a; Roziere et al., 2023). Since our context length is significantly shorter than that of the original LLM, we explore reducing θ for potential performance gains. Our findings show that setting $\theta = 10,000$ with a context length of 1024 enhances performance, so we adopt this configuration moving forward. We note that since we increase the context length (from 512 to 1024), we need to reduce the batch size as well, to not run into memory problems when training. We reduce the batch size by a half and keep the same amount of gradient accumulation steps, which gives us an effective batch size of 128. An ablation of this adaptation is provided in Appendix D.1

Optimiser and Scheduler. Various optimisers and schedulers have been developed to enhance train-

[1]We use the text LM original names for clarity, but note that the actual size will be notably smaller due to reduced vocabulary size, e.g Qwen2.5-0.5B has 358M parameters. Full model sizes can be found in Appendix B.

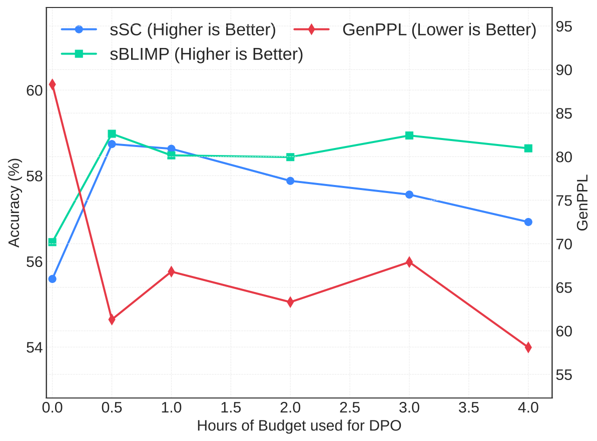

Figure 5: Analysing the optimal part of the 24 hour compute budget that should be used for DPO, with the rest used for pre-training.

ing efficiency, reduce memory usage (Shazeer and Stern, 2018; Dettmers et al., 2022), or accelerate convergence (Pagliardini et al., 2024; Chen et al., 2023). With limited compute, these aspects become especially important. We first consider efficient optimisers, specifically AdamW with fused kernels, and 8-bit AdamW, but observe no notable improvements in batch size or runtime compared to standard AdamW. This could do with the relatively small model size, resulting in a minimal memory footprint of the optimisers. We then compare AdamW with two state-of-the-art optimisers: AdaLomo (Lv et al., 2023) and AdEMAMeix (Pagliardini et al., 2024). Results, presented in Figure 4, suggest that with the original InverseSqrt scheduler used by Hassid et al. (2024), using AdEMAMeix improves validation loss, compared to AdamW, with AdaLomo far behind.

Next, we analyse a cosine decay learning rate scheduler, in place of the original InverseSqrt as this was shown to improve convergence (Loshchilov and Hutter, 2016). We consider the previous optimisers, and provide the validation loss throughout training in Figure 4. We see that this notably improved the loss for AdamW, and slightly harmed results for AdEMAMeix. Overall, AdamW with a cosine schedule provide the best setup, far outperforming the original setup.

4.2 Data

Next, we examine how the training data-mix influences performance in a compute-constrained setting. Specifically, we explore whether diversity in accents, speaking styles, etc. is beneficial and assess if synthetic data can enhance semantic abilities. We provide exact statistics for each dataset in Appendix C.

Model	Data		Metric			
	Div.	Syn.	sBLIMP↑	sSC↑	tSC↑	GenPPL↓
OPT125M	✗	✓	55.28	**55.46**	**75.18**	**96.8**
	✓	✓	55.06	55.00	74.83	116.6
	✗	✗	**55.88**	54.52	70.82	160.3
	✓	✗	55.65	54.78	70.18	172.7
Qwen-0.5B	✗	✓	56.45	**55.59**	**78.01**	**88.3**
	✓	✓	56.17	55.37	77.13	101.3
	✗	✗	**56.60**	53.50	71.14	145.4
	✓	✗	56.10	53.72	70.66	161.8

Table 1: Analysing impact of training data diversity and synthetic data on SLM performance. The default *Slam* recipe does not use diverse data (only Libri-light and LibriSpeech), but uses the synthetic sTinyStories data.

Diverse Data. We begin by examining how dataset diversity impacts model performance. Many leading speech datasets, such as those based on audiobooks (Panayotov et al., 2015; Kahn et al., 2020), consist of relatively clean, single-speaker recordings within a specific content domain. To introduce greater diversity in speaking styles and content, we curate additional datasets, including VoxPopuli (Wang et al., 2021b), Tedlium (Hernandez et al., 2018), PeopleSpeech (Galvez et al., 2021), and SWC (Baumann et al., 2018). For all mentioned datasets, we use the official data cleaning and preprocessing scripts when available. Specifically, for Libri-light, we apply the official Voice Activity Detection model to remove silences and generate smaller audio segments. To evaluate the impact of dataset diversity, we compare the performance of SLMs trained using our best training recipes using a subset of LibriSpeech and Libri-light against all curated datasets. This comparison is conducted for both OPT-125M, which processes a large number of tokens during training, and Qwen-0.5B, which encounters significantly less data due to model size. Results are summarised in Table 1. We observe that dataset diversity has an overall negative effect on model performance. We hypothesise this is due to the models struggling in modelling rich and complex audio under such low compute resources.

Synthetic Data. Recent studies have highlighted the potential of synthetic data generated through Text-to-Speech (TTS) (Cuervo and Marxer, 2024) or direct text-to-unit conversion (Zeng et al., 2024). Hence, we examine the impact of including synthetically generated speech within our constrained compute setup. To do so, we synthesised the

	C (GPU days)	Params	sBLIMP↑	sSC ↑	tSC ↑	GenPPL↓	BLEU↓
TWIST-350M (Hassid et al., 2024)	40*V100	305M	56.20	∅	∅	137.3	3.46
TWIST-1.3B (Hassid et al., 2024)	160*V100	1B	57.00	52.4	70.6	131.8	3.20
TWIST-7B (Hassid et al., 2024)	∅	7B	59.00	55.3	74.1	93.7	3.06
TWIST-13B (Hassid et al., 2024)	∅	13B	59.20	55.4	76.4	∅	∅
Scaled Optimal (Cuervo and Marxer, 2024)	∅	823M	**61.3**	**56.7**	**78.0**	∅	∅
Predicted Optimal (Cuervo and Marxer, 2024)	1*A5000	78M	56.85	54.09	70.49	∅	∅
TWIST-350M (Original recipe)	1*A5000	305M	51.52 ±.19	53.65 ±.57	68.80 ±.47	259.2 ±6.7	3.26 ±.46
TWIST-350M + sTinyStories	1*A5000	305M	51.21 ±.26	54.17 ±.54	72.40 ±.18	159.0 ±6.0	4.18 ±.24
Slam (-DPO) (ours)	1*A5000	358M	56.45 ±.17	55.59 ±.30	78.01 ±.27	88.3 ±1.0	3.47 ±.17
Slam (ours)	1*A5000	358M	**58.86** ±.20	**58.04** ±.51	**82.04** ±.21	**62.8** ±4.1	3.88 ±.11

Table 2: Comparing *slamming* to leading SLMs, and predicted optimal performance for the compute. We also consider TWIST-350M using our code and compute budget, but with the original training recipe. ± indicates distance to min/max of 3 seeds. BLEU is Auto-BLEU.

TinyStories dataset (Eldan and Li, 2023) using a single-speaker TTS model (Wang et al., 2021a), as it is computationally efficient. Additionally, prior research has shown that HuBERT units largely remove speaker information (Maimon and Adi, 2023). TinyStories has been demonstrated to enhance text LM performance and improve SLMs (Cuervo and Marxer, 2024). Results are presented in Table 1. Results indicate that incorporating such synthetic data into the training data-mix significantly boosts both modelling and generative performance metrics, across all evaluated setups. We also consider adding the synthetic data to the original TWIST recipe, and the results in the bottom of Table 2 suggests that while this helps with semantic metrics, it is far from enough without other optimisations we introduced. As a further ablation, we assess the performance of SLM when trained exclusively on synthetic data. Results suggest, perhaps unsurprisingly, this leads to a significant drop in performance relative to our baseline model, which uses both real and synthetic data. Specifically, the model trained only on synthetic data scores 52.35 on sBLIMP, compared to 56.45 for the baseline, and exhibits a notably higher validation loss on real data (2.8 vs. 1.65). We observe this across all datasets, and specifically with our best mixture Libri-Light, LibriSpeech and sTinyStories, Qwen-0.5B outperforms OPT-125M so we continue with it to the final stages. These findings reinforce the importance of incorporating both real and synthetic data during training.

4.3 Text Interleaving

Several recent SLMs combine both speech and text modalities, either predicting both simultaneously (Défossez et al., 2024; Fang et al., 2024; Xie and Wu, 2024) or training on interleaved data (Nguyen et al., 2025; Zeng et al., 2024). Beyond enhancing cross-modal abilities, this has been shown to improve the semantic capabilities of SLMs, even in speech-only evaluations. Building on these studies, we investigate whether speech-text interleaving can enhance semantic ability in speech-only tasks, even under strict computational constraints.

For this we use Whisper-large-v3-turbo to get aligned transcriptions of our data, except sTinyStories for which we get alignment from the TTS. We follow Zeng et al. (2024) by selecting speech spans with length from a Poisson distribution with $\lambda = 10$ totalling 30% of the interleaved data. Following Nguyen et al. (2025) we train with balanced batches with respect to token count between text data, speech data and interleaved data. We use a subset of RedPajama (Weber et al., 2024) filtered by Gopher (Rae et al., 2021) rules as our text data.

The SLM trained with interleaving with the exact same setup as the speech only variant slightly underperformed compared to the speech only. We report results as the mean of three training runs. Specifically, it achieved tSC of 73.36 (compared to 78.01 for the speech only equivalent), sSC of 55.76 (vs 55.59) and sBLIMP of 55.71 (vs 56.45). We note that the interleaved SLM has much larger vocabulary which in turn means that the model has more parameters ($\sim 500M$ vs $\sim 360M$), which in turn means that each update step takes longer. For our budget the interleaved model only performed $\sim 11k$ steps vs $\sim 18k$ for speech only. Furthermore, out of all training tokens only about 40% are speech tokens in the interleaved setting. This could perhaps explain the slightly worse performance, and we leave for future work to find the minimal compute budget to benefit from text-interleaving.

	GPUs	Params	Num tokens	sBLIMP↑	sSC ↑	tSC ↑	GenPPL↓	BLEU↓	GPTScore↑
Speech only pre-training									
GSLM (Lakhotia et al., 2021)	8*V100	100M	1B	54.2	53.3	66.6	∅	∅	∅
SyllableLM (Baade et al., 2024)	4*A40	300M	16B	63.7	∅	75.4	∅	∅	∅
TWIST-350M (Hassid et al., 2024)	8*V100	305M	10.8B	56.20	∅	∅	137.3	3.46	∅
TWIST-1.3B (Hassid et al., 2024)	32*V100	1B	10.8B	57.00	52.4	70.6	131.8	3.20	1.82
TWIST-7B (Hassid et al., 2024)	32*V100	7B	36B	59.00	55.3	74.1	93.74	3.06	2.71
TWIST-13B (Hassid et al., 2024)	32*V100	13B	36B	59.20	55.4	76.4	∅	∅	∅
Cuervo and Marxer (2024)	∅	823M	82B	**61.3**	56.7	78.0	∅	∅	∅
Moshi (Défossez et al., 2024)	?*H100	7B	∅	58.9	**58.7**	**81.8**	∅	∅	∅
SpiritLM (Nguyen et al., 2025)	64*A100	7B	100B	58.0	54.8	72.9	∅	∅	∅
Joint speech-text pre-training / preference optimisation									
Zeng et al. (2024)	∅	9B	~1T	∅	**62.4**	82.9	∅	∅	∅
Moshi (Défossez et al., 2024)	?*H100	7B	~720B	58.8	60.8	83.0	∅	∅	∅
SpiritLM (Nguyen et al., 2025)	64*A100	7B	100B	58.3	61.0	82.9	∅	∅	∅
AlignSLM-1.3B (Lin et al., 2024)	64*A100	1B	10.8B + ~158B	59.8	55.0	80.0	∅	∅	2.43
AlignSLM-7B (Lin et al., 2024)	64*A100	7B	36B + ~158B	**62.3**	61.1	**86.8**	∅	∅	**3.50**
Slam (-DPO)	2*A100	358M	16.7B	58.53	58.15	80.71	67.3	3.25	∅
Slam	1*A5000	358M	1.4B + 5M	58.86	58.04	82.04	62.8	3.88	2.09
Slam (scaled)	2*A100	358M	16.7B + 9M	61.11	61.30	84.18	46.6	3.75	2.69
Slam (large)	2*A100	1.3B	6.1B + 9M	**61.43**	**61.52**	**85.30**	**41.2**	3.89	**2.79**

Table 3: Analysing the effect of scaling up compute for *Slam*. # tokens refers to total, not unique, tokens used for training (estimated from provided information). We separately mark DPO tokens with a +. BLEU is Auto-BLEU.

4.4 Synthetic Data Preference Optimisation

Preference optimisation methods have been shown to enhance the performance of text LLMs (Ouyang et al., 2022) and, more recently, SLMs (Lin et al., 2024). With preference optimisation, we aim to train our model to generate outputs that better align with a specified reward function or preference set.

We evaluate how preference optimisation affects SLM performance while considering our constrained computational budget. Using an *off-policy* approach with pre-generated preference data, we apply DPO to enhance training efficiency. Specifically, we synthetically generate the SWAG (Zellers et al., 2018) text corpus for evaluating semantic knowledge. SWAG consists of text prefixes paired with multiple possible suffixes, where only one is semantically plausible. For preference data, we use the first sentence as the prompt, the correct suffix as the positive continuation, and a randomly chosen incorrect suffix as the rejected continuation. To ensure quality, we filter out samples with repetitive patterns, identified by an auto-BLEU score above 0.3. We generate all recordings using Kokoro TTS (Hexgrad, 2025), incorporating four speakers (two male and two female), evenly split between British and American accents. This process results in a total of 47k SWAG preference pairs.

For DPO we use $\beta = 0.1$ (see Appendix A for full hyperparameters). In initial tests, we observe that after DPO training, the model shows increased likelihood at the cost of repeated patterns, a known issue with DPO (Lanchantin et al., 2025). To address this, we apply a repetition penalty with a factor of 1.1, following the approach of Keskar et al. (2019), and find that it helps mitigate the problem. Future work could explore alternative solutions, such as proposed by Lanchantin et al. (2025).

We begin by examining how the allocation of budget for DPO impacts performance, particularly when it comes at the cost of a shorter pre-training phase. Figure 5 depicts the results. We observe significant improvements across all metrics when applying DPO for at least 30 minutes compared to not using DPO at all. However, allocating a higher proportion of the budget to DPO does not yield further gains and can even degrade model performance. Thus we stick to 30 minutes out of 24 hours for DPO, using the rest for pre-training.

5 Final Recipe

Building on these empirical findings, we develop the final *Slam* recipe. Using it, we train SLMs based on Qwen2.5-0.5B. We then compare *Slam* to the TWIST model family across various sizes: 350M, 1.3B, 7B, and 13B. We also present results for TWIST-350M using our computational constraints but following TWIST's original training recipe, along with our synthetic data. Finally, we report results for the top-performing model from (Cuervo and Marxer, 2024), including their predicted optimal performance under our compute budget based on SLM scaling laws. Results are reported in Table 2. The results indicate that *Slam* delivers performance that is either superior or on par with baseline models while requiring significantly fewer computational resources (e.g., a single A5000 for a day compared to 160 days on a V100). Transcribed generated examples by *Slam* can be seen in Appendix D.4.

To show that the *Slam* models do not overfit a single domain (audiobooks/stories), we provide results for GenPPL on a different domain. This can be seen in Appendix D.3.

We further evaluate the quality of the generated audio using Mosnet (Cooper et al., 2022), similarly to Align-SLM. Results are presented in Appendix D.2. As the quality of the generated audio is mainly affected by the vocoder, which is identical across evaluated methods, results are comparable. Interestingly, TWIST 1.3B and TWIST 7B achieve slightly worse scores.

6 Increasing Compute

Similarly to Geiping and Goldstein (2023), we analyse whether the proposed approach holds well also in increased compute budget. We opt for 48 hours on 2 A100 GPUs as a reasonable academic budget for larger scale tests, and represents $\sim$ 10 times more compute than the *Slamming* setting. We use exactly the same *Slam* recipe for more steps, and increase the batch size times 2. We provide the full results in Table 3. We note that the performance continues to improve across all metrics, also outperforming methods which have far larger compute scales. We note that DPO training on synthetic data for 2 epochs, notably boosts performance. Transcribed generated examples by *Slam* (scaled) can be seen in Appendix D.4

We also wish to assess whether our suggested recipe holds for larger models, thus we evaluate training a larger Qwen2.5 text LM as the base model. We use $\text{Qwen2.5} - 1.5\text{B}$ for the same compute budget as above - i.e two A100 GPUs for 48 hours. All training details are identical, but of course the larger model was trained for less steps (and tokens). We provide results from this model, denoted *Slam* (large) in Table 3. Results show that this model even outperforms the smaller model for the same compute budget. This demonstrates that the *Slam* recipe holds for larger models, and reiterates the importance of quality models even at the expense of less training tokens for this setup.

7 Limitations

While the SLMs trained under *Slamming* compute budget performed notably well compared to other SLMs trained with much more compute they might perform less well in other areas. For instance, evaluating their abilities on acoustic or prosodic elements as in SALMon (Maimon et al., 2024) could show further challenges of low resource settings.

Furthermore, we focus in this study on the well used HuBERT (Hsu et al., 2021) model as a tokeniser, and while we do not make any adjustments specifically for it, future work might wish to investigate our cramming approach with new tokenisers, such as Mimi (Défossez et al., 2024) and SylBoost (Baade et al., 2024).

8 Conclusion

In this work we show that training high quality SLMs with a very modest compute budget, is feasible. We give these main guidelines:

1. **Do not skimp on the model** - not all model families are born equal and the TWIST initialisation exaggerates this, thus it is worth selecting a stronger / bigger text-LM even if it means less tokens. we found Qwen2.5 to be a good choice.
2. **Utilise synthetic training data** - pre-training on data generated with TTS helps a lot.
3. **Go beyond next token prediction** - we found that DPO boosts performance notably even when using synthetic data, and as little as 30 minutes training massively improves results.
4. **Optimise hyper-parameters** - as researchers we often dis-regard this stage, yet we found that tuning learning rate schedulers and optimising code efficiency can improve results notably.

We hope that these insights, and open source resources will be of use to the community in furthering SLM research.

Ethical Statement

The broader impact of this study is, as in any generative model, the development of a high quality and natural speech synthesis. We hope that allowing training SLMs under low-resource settings, and open sourcing resources to aid this goal, will have a positive impact on inclusivity and accessibility of SLM research beyond well funded labs.

Acknowledgements. This research work was supported by ISF grant 2049/22.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, et al. 2025. Smollm2: When smol goes big–data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. 2024. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pages 929–947.

Alan Baade, Puyuan Peng, and David Harwath. 2024. Syllablelm: Learning coarse semantic units for speech language models. *arXiv preprint arXiv:2410.04029*.

Timo Baumann, Arne Köhn, and Felix Hennig. 2018. The spoken wikipedia corpus collection: Harvesting, alignment and an application to hyperlistening. *Language Resources and Evaluation*, 53:303 – 329.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. 2023. Audiolm: a language modeling approach to audio generation. *IEEE/ACM transactions on audio, speech, and language processing*, 31:2523–2533.

Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V. Le. 2023. Symbolic discovery of optimization algorithms. *Preprint*, arXiv:2302.06675.

Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. 2023. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*.

Erica Cooper, Wen-Chin Huang, Tomoki Toda, and Junichi Yamagishi. 2022. Generalization ability of mos prediction networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8442–8446. IEEE.

Santiago Cuervo and Ricard Marxer. 2024. Scaling properties of speech language models. *arXiv preprint arXiv:2404.00685*.

Wenqian Cui, Dianzhi Yu, Xiaoqi Jiao, Ziqiao Meng, Guangyan Zhang, Qichao Wang, Yiwen Guo, and Irwin King. 2024. Recent advances in speech language models: A survey. *arXiv preprint arXiv:2410.03751*.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *Preprint*, arXiv:2307.08691.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. 2024. Moshi: a speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2022. 8-bit optimizers via block-wise quantization. *Preprint*, arXiv:2110.02861.

Ewan Dunbar, Mathieu Bernard, Nicolas Hamilakis, Tu Anh Nguyen, Maureen De Seyssel, Patricia Rozé, Morgane Rivière, Eugene Kharitonov, and Emmanuel Dupoux. 2021. The zero resource speech challenge 2021: Spoken language modelling. *arXiv preprint arXiv:2104.14700*.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.

Avishai Elmakies, Omri Abend, and Yossi Adi. 2025. Unsupervised speech segmentation: A general approach using speech language models. *Preprint*, arXiv:2501.03711.

Qingkai Fang, Shoutao Guo, Yan Zhou, Zhengrui Ma, Shaolei Zhang, and Yang Feng. 2024. Llama-omni: Seamless speech interaction with large language models. *arXiv preprint arXiv:2409.06666*.

Daniel Galvez, Greg Diamos, Juan Manuel Ciro Torres, Juan Felipe Cerón, Keith Achorn, Anjali Gopi, David Kanter, Max Lam, Mark Mazumder, and Vijay Janapa Reddi. 2021. The people's speech: A large-scale diverse english speech recognition dataset for commercial usage. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Jonas Geiping and Tom Goldstein. 2023. Cramming: Training a language model on a single gpu in one day. In *International Conference on Machine Learning*, pages 11117–11143. PMLR.

Albert Gu, Karan Goel, and Christopher Ré. 2021. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*.

Habib Hajimolahoseini, Omar Mohamed Awad, Walid Ahmed, Austin Wen, Saina Asani, Mohammad Hassanpour, Farnoosh Javadi, Mehdi Ahmadi, Foozhan Ataiefard, Kangling Liu, et al. 2023. Swiftlearn: A data-efficient training method of deep learning models using importance sampling. *arXiv preprint arXiv:2311.15134*.

Michael Hassid, Tal Remez, Tu Anh Nguyen, Itai Gat, Alexis Conneau, Felix Kreuk, Jade Copet, Alexandre Defossez, Gabriel Synnaeve, Emmanuel Dupoux, et al. 2024. Textually pretrained speech language models. *Advances in Neural Information Processing Systems*, 36.

François Hernandez, Vincent Nguyen, Sahar Ghannay, Natalia Tomashenko, and Yannick Esteve. 2018. Tedlium 3: Twice as much data and corpus repartition for experiments on speaker adaptation. In *Speech and Computer: 20th International Conference, SPECOM 2018, Leipzig, Germany, September 18–22, 2018, Proceedings 20*, pages 198–208. Springer.

Hexgrad. 2025. Kokoro-82m (revision d8b4fc7).

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing*, 29:3451–3460.

Peter Izsak, Moshe Berchansky, and Omer Levy. 2021. How to train bert with an academic budget. *arXiv preprint arXiv:2104.07705*.

Shengpeng Ji, Yifu Chen, Minghui Fang, Jialong Zuo, Jingyu Lu, Hanting Wang, Ziyue Jiang, Long Zhou, Shujie Liu, Xize Cheng, et al. 2024. Wavchat: A survey of spoken dialogue models. *arXiv preprint arXiv:2411.13577*.

Jacob Kahn, Morgane Riviere, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, et al. 2020. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7669–7673. IEEE.

Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee, Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen, et al. 2019. A study of bfloat16 for deep learning training. *arXiv preprint arXiv:1905.12322*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Eugene Kharitonov et al. 2021. Text-free prosody-aware generative spoken language modeling. *arXiv preprint arXiv:2109.03264*.

Felix Kreuk et al. 2021. Textless speech emotion conversion using decomposed and discrete representations. *arXiv preprint arXiv:2111.07402*.

Kushal Lakhotia, Eugene Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Abdelrahman Mohamed, et al. 2021. On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 9:1336–1354.

Jack Lanchantin, Angelica Chen, Shehzaad Dhuliawala, Ping Yu, Jason Weston, Sainbayar Sukhbaatar, and Ilia Kulikov. 2025. Diverse preference optimization. *arXiv preprint arXiv:2501.18101*.

Siddique Latif, Moazzam Shoukat, Fahad Shamshad, Muhammad Usama, Yi Ren, Heriberto Cuayáhuitl, Wenwu Wang, Xulong Zhang, Roberto Togneri, Erik Cambria, et al. 2023. Sparks of large audio models: A survey and outlook. *arXiv preprint arXiv:2308.12792*.

Junyan Li, Delin Chen, Tianle Cai, Peihao Chen, Yining Hong, Zhenfang Chen, Yikang Shen, and Chuang Gan. 2024. Flexattention for efficient high-resolution vision-language models. In *European Conference on Computer Vision*, pages 286–302. Springer.

Xiang Li, Yiqun Yao, Xin Jiang, Xuezhi Fang, Xuying Meng, Siqi Fan, Peng Han, Jing Li, Li Du, Bowen Qin, et al. 2023. Flm-101b: An open llm and how to train it with $100 k budget. *arXiv preprint arXiv:2309.03852*.

Guan-Ting Lin, Prashanth Gurunath Shivakumar, Aditya Gourav, Yile Gu, Ankur Gandhe, Hung-yi Lee, and Ivan Bulyko. 2024. Align-slm: Textless spoken language models with reinforcement learning from ai feedback. *arXiv preprint arXiv:2411.01834*.

Andy T Liu, Yi-Cheng Lin, Haibin Wu, Stefan Winkler, and Hung-yi Lee. 2024a. Efficient training of self-supervised speech foundation models on a compute budget. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 961–968. IEEE.

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, et al. 2024b. Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. In *Forty-first International Conference on Machine Learning*.

Team LLama. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

I Loshchilov. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Ilya Loshchilov and Frank Hutter. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Kai Lv, Hang Yan, Qipeng Guo, Haijun Lv, and Xipeng Qiu. 2023. Adalomo: Low-memory optimization with adaptive learning rate. *arXiv preprint arXiv:2310.10195*.

Xingtai Lv, Ning Ding, Kaiyan Zhang, Ermo Hua, Ganqu Cui, and Bowen Zhou. 2024. Scalable efficient training of large language models with low-dimensional projected attention. *arXiv preprint arXiv:2411.02063*.

Gallil Maimon and Yossi Adi. 2023. Speaking style conversion in the waveform domain using discrete self-supervised units. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8048–8061.

Gallil Maimon, Amit Roth, and Yossi Adi. 2024. A suite for acoustic language model evaluation. *arXiv preprint arXiv:2409.07437*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*.

Aashiq Muhamed, Oscar Li, David Woodruff, Mona Diab, and Virginia Smith. 2024. Grass: Compute efficient low-memory llm training with structured sparse gradients. *arXiv preprint arXiv:2406.17660*.

Eliya Nachmani et al. 2024. Spoken question answering and speech continuation using spectrogram-powered llm. In *The Twelfth International Conference on Learning Representations*.

Evgeny Hershkovitch Neiterman and Gil Ben-Artzi. 2024. Layerdropback: A universally applicable approach for accelerating training of deep networks. *arXiv preprint arXiv:2412.18027*.

Tu Anh Nguyen, Benjamin Muller, Bokai Yu, Marta R Costa-Jussa, Maha Elbayad, Sravya Popuri, Christophe Ropers, Paul-Ambroise Duquenne, Robin Algayres, Ruslan Mavlyutov, et al. 2025. Spirit-lm: Interleaved spoken and written language model. *Transactions of the Association for Computational Linguistics*, 13:30–52.

Tu Anh Nguyen et al. 2022. Generative spoken dialogue language modeling. *arXiv preprint arXiv:2203.16502*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Matteo Pagliardini, Pierre Ablin, and David Grangier. 2024. The ademamix optimizer: Better, faster, older. *Preprint*, arXiv:2409.03137.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Se Jin Park, Julian Salazar, Aren Jansen, Keisuke Kinoshita, Yong Man Ro, and RJ Skerry-Ryan. 2024. Long-form speech generation with spoken language models. *arXiv preprint arXiv:2412.18603*.

Jing Peng, Yucheng Wang, Yu Xi, Xu Li, Xizhuo Zhang, and Kai Yu. 2024a. A survey on speech large language models. *arXiv preprint arXiv:2410.18908*.

Yifan Peng et al. 2024b. Mslm-s2st: A multitask speech language model for textless speech-to-speech translation with speaker style preservation. *arXiv preprint arXiv:2403.12408*.

Adam Polyak et al. 2021. Speech resynthesis from discrete disentangled self-supervised representations. *arXiv preprint arXiv:2104.00355*.

Sravya Popuri et al. 2022. Enhanced direct speech-to-speech translation using self-supervised pre-training and data augmentation. *arXiv preprint arXiv:2204.02967*.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Ankit Singh Rawat, Veeranjaneyulu Sadhanala, Afshin Rostamizadeh, Ayan Chakrabarti, Wittawat Jitkrittum, Vladimir Feinberg, Seungyeon Kim, Hrayr Harutyunyan, Nikunj Saunshi, Zachary Nado, et al. 2024. A little help goes a long way: Efficient llm training by leveraging small lms. *arXiv preprint arXiv:2410.18779*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Jay Shah, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. 2024. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. *arXiv preprint arXiv:2407.08608*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. *Preprint*, arXiv:1804.04235.

Li Shen, Yan Sun, Zhiyuan Yu, Liang Ding, Xinmei Tian, and Dacheng Tao. 2023. On efficient training of large-scale deep learning models: A literature review. *arXiv preprint arXiv:2304.03589*.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun MA, and Chao Zhang. 2024. SALMONN: Towards generic hearing abilities for large language models. In *The Twelfth International Conference on Learning Representations*.

Changhan Wang, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Ann Lee, Peng-Jen Chen, Jiatao Gu, and Juan Pino. 2021a. fairseq s^2: A scalable and integrable speech synthesis toolkit. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 143–152.

Changhan Wang, Morgane Riviere, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. 2021b. VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. 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 993–1003, Online. Association for Computational Linguistics.

Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, et al. 2023. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*.

Jiachen T Wang, Tong Wu, Dawn Song, Prateek Mittal, and Ruoxi Jia. 2024. Greats: Online selection of high-quality data for llm training in every iteration. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, et al. 2024. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. *arXiv preprint arXiv:2412.13663*.

Maurice Weber, Daniel Y. Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. 2024. Redpajama: an open dataset for training large language models. *NeurIPS Datasets and Benchmarks Track*.

Zhifei Xie and Changqiao Wu. 2024. Mini-omni: Language models can hear, talk while thinking in streaming. *arXiv preprint arXiv:2408.16725*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024a. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Dongchao Yang et al. 2023. Uniaudio: An audio foundation model toward universal audio generation. *arXiv preprint arXiv:2310.00704*.

Dongchao Yang et al. 2024b. Uniaudio 1.5: Large language model-driven audio codec is a few-shot audio task learner. *arXiv preprint arXiv:2406.10056*.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Aohan Zeng, Zhengxiao Du, Mingdao Liu, Lei Zhang, Shengmin Jiang, Yuxiao Dong, and Jie Tang. 2024. Scaling speech-text pre-training with synthetic interleaved data. *arXiv preprint arXiv:2411.17607*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

A Full *Slam* Recipe

We provide below the full training recipe, including hyperparameters for the best, *Slam* recipe. In Table 4 we see the *Slam* (-DPO) pre-training recipe and in Table 5 we see the *Slam* DPO training recipe. Table 6 provides the sampling hyper-parameters used for calculating the generative metrics. Note that some of the generated samples in the demo page were created with a higher maximum token limit.

Table 4: *Slam* (-DPO) pre-training recipe.

Parameter	Value
Text Base Model	Qwen2.5-0.5B
TWIST initialisation	True
Data	Librilight, Librispeech, sTinyStories
Train Time	23.5 hours $\simeq$ 17625 steps
RoPE theta	10000
Context length	1024
Per device Batch Size	8
Gradient Accumulation	16
Base Learning Rate	$1e-3$
Warmup Ratio	1%
Optimizer	AdamW
LR Scheduler	cosine with min $5e-5$
Max Grad Norm	0.5
Dtype	bfloat16

Table 5: *Slam* DPO training recipe.

Parameter	Value
Initial Model	*Slam* (-DPO)
Data	SpokenSwag with auto-bleu≤ 0.3
Train Time	0.5 hour $\simeq$ 813 steps
RoPE theta	10000
Context length	1024
Per device Batch Size	4
Gradient Accumulation	16
Base Learning Rate	$5e-5$
Optimizer	AdamW
Learning Rate Scheduler	inverse sqrt
Max Grad Norm	0.5
Dtype	bfloat16
DPO β	0.1

Table 6: *Slam* sampling parameters.

Parameter	Value
Temperature	0.8
Top-K	25
Max New Tokens	150
Repetition Penalty	1.1

Table 7: Model names and parameter counts after changing vocabulary to speech only units (500).

Model Name	Params
MobileLLM-125M (Liu et al., 2024b)	106,492,608
MobileLLM-350M (Liu et al., 2024b)	315,117,120
OPT-125M (Zhang et al., 2022)	87,015,936
OPT-350M (Zhang et al., 2022)	305,714,176
QWEN2.5-0.5B (Yang et al., 2024a)	358,347,904
SmolLM2-135M (Allal et al., 2025)	106,492,608
SmolLM2-360M (Allal et al., 2025)	315,117,120
Pythia-160M (Biderman et al., 2023)	85,827,072
Pythia-410M (Biderman et al., 2023)	303,339,520

B Model Sizes

As mentioned, we use the original names of the text LMs used for clarity and consistency, but note that the actual parameter counts after resizing the vocabulary to speech-units only can be very different. In Table 7 we provide an extensive list of models and sizes.

C Dataset Statistics

We use and synthesise several datasets. In this section we give exact details of number of samples, splits used, domains etc.

For pre-training we use Libri-Light (Kahn et al., 2020) and LibriSpeech (Panayotov et al., 2015). For Libri-Light we randonly select one percent of samples as validation, whereas for LibriSpeech we use the original *dev-clean* and *dev-other* splits. Both of these datasets are English speech only, focused in the audio-book domain. We also synthesise sTinyStories for pre-training which consists of synthetically generated English short stories. We use the official train split for training. Full dataset sizes are in Table 8.

We also investigate diverse datasets for pre-training: SWC (Baumann et al., 2018), Tedlium (Hernandez et al., 2018), PeopleSpeech (Galvez et al., 2021) and VoxPopuli (Wang et al., 2021b). We only take English subsets for all datasets, yet they can still contain diverse accents. These datasets are in the following domains SWC - read Wikipedia articles, Tedlium - short lectures, PeopleSpeech - diverse data including many local council gatherings etc, VoxPopuli - from European Parliament meetings. For SWC specifically, we use the text alignment to create chunks, remove silence from the audio and remove mis-aligned chunks. We use full training splits where provided, otherwise splitting 99% for training. The dataset sizes are described in Table 8.

Table 8: Training set size for used datasets.

Dataset	Hours	Tokens
Libri-Light (Kahn et al., 2020)	$50K$	$3.5B$
LibriSpeech (Panayotov et al., 2015)	960	$67M$
SWC (Baumann et al., 2018)	750	$19M$
Tedlium (Hernandez et al., 2018)	$1.6K$	$110M$
PeopleSpeech (Galvez et al., 2021)	$7K$	$480M$
VoxPopuli (Wang et al., 2021b)	$24K$	$1.64B$
sTinyStories	$30K$	$2.2B$

For DPO we synthesise SpokenSwag based on the SWAG (Zellers et al., 2018) dataset. We use only the official train set and filter only the gold standard labels. We end up with 47k sample pairs which end up to be $\sim 4.5M$ tokens.

D Additional Results

D.1 Context Length and Batch Size Ablation

In Table 9 we see results for ablations of context length and effective batch size

Table 9: Performance on sBlimp, tStoryCloze (TSC), sStoryCloze (SSC) and validation loss across different context lengths and effective batch sizes. using Qwen2.5-0.5B

Context	BS	sBLIMP↑	tSC↑	sSC↑	Val loss↓
512	128	56.13	76.91	55.53	1.67
512	256	**56.56**	77.49	**56.33**	1.66
1024	256	56.43	**78.88**	55.69	1.65
1024	128	56.45	78.01	55.59	**1.64**

D.2 MOS Proxy Results

For completeness we also provide MOS proxy results for our models compared to TWIST and Align-SLM models. We follow a similar setup to Lin et al. (2024) and use MOSnet to test the audio's generation quality of our models. It is important to note that we use the same vocoder as TWIST and Align-SLM. The results can be seen in Table 10.

D.3 GenPPL on Different Domain

In order to evaluate the generalisability of our approach to diverse domains, we calculate GenPPL for a dataset from a different domain. We compare our results to TWIST of various sizes, which were trained on this exact dataset (perhaps even overlapping samples as no official training set was published). We use the same setup for GenPPL as described in section 3, but we use People Speech test set (Galvez et al., 2021) as prompts. Results in Table 11, show that *Slam* performs comparably or better to TWIST models of larger scale and more train compute, despite the fact they were explicitly trained on this dataset. These results highlight the efficacy of the *Slam* recipe beyond a single domain.

Table 10: MOSnet scores for various models.

Model	MOSnet ↑
Ground Truth	4.28
TWIST-350M (Hassid et al., 2024)	4.07
TWIST-1.3B (Hassid et al., 2024)	3.83
TWIST-7B (Hassid et al., 2024)	3.85
Align-SLM-1.3B (Lin et al., 2024)	4.05
Align-SLM-7B (Lin et al., 2024)	4.09
Slam	**4.11**
Slam (scaled)	4.07
Slam (large)	4.05

Table 11: GenPPL results on the People Speech Dataset.

Model	GenPPL ↓	BLEU ↓
TWIST-350M (Hassid et al., 2024)	162.89	2.36
TWIST-1.3B (Hassid et al., 2024)	155.51	2.86
TWIST-7B (Hassid et al., 2024)	128.74	2.72
Slam	171.84	4.89
Slam (scaled)	128.78	3.99
Slam (large)	**117.49**	3.72

D.4 Text Generation Examples

For completeness we provide some transcriptions for some of the generated examples (generated by *Slam* and *Slam* (Scaled)). The prompts and the transcriptions of the generated samples can be seen in Table 12.

E AI Tool Usage

AI based tools may have been used in writing parts of the code for this study, or para-phrasing some of the writing within the paper, yet all the content was thoroughly checked by the authors, with these only being used as assistive tools.

Table 12: transcriptions of generated continuations from Slam and Slam Scaled for various prompts.

Prompt	Slam	Slam Scaled
Johnny found an old coin on the subway floor…	and decided to bring it home. Then he came up by a different subway floor, which opened into a courtyard. Johnny picked up the coin and threw it at the doorstep of the courtyard.	He picked it up and looked at the size and shape of the coin. It must be valuable, remembering his promise to Johnny, had it put very carefully in his pocket. But now he felt as if somebody had given him a special coin in return.
I am a real boy he said	He looked at Sarah Weinfeild and smiled. It was quite impossible for Sarah to look at him any longer, and had no suspicions about his feelings for a moment.	So off he went to school in the spring time, sitting on the soft grass near the trees where the rain fell in torrents. As the sun began to set, the birds sang their lovely songs and the children clapped their hands with Joe and Jack Pensilvenya.
I had a great day today playing	I saw a little boy playing too. He said hello to me, I went to say hello to you. I said hello back. I thought he was very nice, I asked him if he wanted to play with me? He said yes, and they played together telling jokes and laughter in the quiet little world. They had a great time together, and when it was time to go home I said bye to the little boy and went to sleep. end.	I love playing the flute, and almost forgot my troubles in the warm sunshine as I set the flute for miles around in the air practicing my favorite songs. My flute singing out in answer to the beautiful sounds from the world around me. At length, the song ended and I stood up and clapped my hands, then we passed on through the forest, over the meadow and along the river bank until we came to a beautiful valley.