Recipes for Sequential Pre-training of Multilingual Encoder and Seq2Seq Models

Saleh Soltan* Alexa AI, Amazon ssoltan@amazon.com

Andy Rosenbaum* Alexa AI, Amazon andros@amazon.com

Tobias Falke Alexa AI, Amazon falket@amazon.de

Qin Lu Alexa AI, Amazon luqn@amazon.com

Anna Rumshisky Alexa AI, Amazon Univ. of Massachusetts Lowell arrumshi@amazon.com

Wael Hamza Alexa AI, Amazon waelhamz@amazon.com

Abstract

Pre-trained encoder-only and sequence-to-sequence (seq2seq) models each have advantages, however training both model types from scratch is computationally expensive. We explore recipes to improve pre-training efficiency by initializing one model from the other. (1) Extracting the encoder from a seq2seq model, we show it under-performs a Masked Language Modeling (MLM) encoder, particularly on sequence labeling tasks. Variations of masking during seq2seq training, reducing the decoder size, and continuing with a small amount of MLM training do not close the gap. (2) Conversely, using an encoder to warm-start seq2seq training, we show that by unfreezing the encoder partway through training, we can match task performance of a from-scratch seq2seq model. Overall, this two-stage approach is an efficient recipe to obtain both a multilingual encoder and a seq2seq model, matching the performance of training each model from scratch while reducing the total compute cost by 27%.

1 Introduction and Related Work

Transformer-based Pre-trained Language Models (PLMs) have become the main building blocks when creating models for most Natural Language Processing (NLP) tasks. PLMs come in three main architectures: decoder-only (e.g. GPT), sequence-to-sequence (seq2seq, e.g. BART, T5), and encoder-only (e.g. BERT). Multilingual models such as XLM-RoBERTa (encoder-only) and mBART/mT5 (seq2seq) are also common.

Raffel et al. (2020b) showed that seq2seq models can perform many NLP tasks on par with similarly-sized encoder-only models trained via Masked Language Modeling (MLM) by framing tasks such a sentence classification or sequence labeling as text generation. However, encoder models remain more efficient at inference for sequence labeling tasks like Named Entity Recognition (NER) and Part-of-Speech tagging (POS): an encoder can label all words in the sequence with a single forward pass, while a seq2seq model must generate each word's label autoregressively.

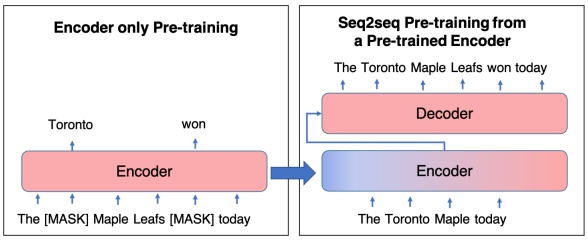


Figure 1: Two-stage seq2seq pre-training. First (left), we train the encoder via Masked Language Modeling (MLM). Second (right), we attach a randomly initialized decoder to the pre-trained MLM encoder, and train on the same data with de-noising objective. The encoder may remain frozen for part or all of the second stage.

Motivated by the need for both an encoder model for efficient sequence labeling and a seq2seq model for generative tasks like semantic parsing and summarization, we explore recipes to pre-train both models. Compared to training each model from scratch, we propose two sequential training recipes which reduce the total compute cost (Section 2.1.6).

The first recipe is to extract the encoder of a seq2seq model as proposed in Ni et al. (2022). Although it performs well on classification tasks, we show that the encoder from seq2seq under-performs a from-scratch encoder on sequence labeling tasks. Variations of masking during seq2seq training and reducing the decoder size do not provide a consistent benefit to the encoder. We also explore continuing training the extracted encoder on MLM for a small number of updates. However, we show it cannot consistently close the gap in performance across different datasets.

The second recipe is to warm-start seq2seq pre-training with an encoder pre-trained via MLM (Fig-

* Equal Contribution.

Findings of the Association for Computational Linguistics: ACL 2023, pages 9380–9394

ure 1). Rothe et al. (2020) proposed a similar idea for fine-tuning. AlexaTM 20B and AlexaTM 5B applied this recipe for pre-training, by warm-starting with Alexa Teacher Model encoders (Soltan et al., 2022; Rosenbaum et al., 2022b; FitzGerald et al., 2022). We add the novelty of comparing to a seq2seq model pre-trained from scratch with the same data and codebase. First, we observe that if the encoder is frozen the whole time, the model under-performs a from-scratch seq2seq model on semantic parsing and summarization tasks. While cross-attention fusion across different layers of the encoder reduces the performance gap, we find that we can match performance of a from-scratch model by using standard cross-attention and unfreezing the encoder partway through training.

Overall, the second recipe demonstrates a viable approach for efficient pre-training of both a multilingual encoder and a multilingual seq2seq model, matching the performance of training each model from scratch, while using 27% less total compute.

See Appendix A for additional related work.

2 Pre-Training Setup

We describe our pre-training objectives, models, datasets, two recipes for initializing one model type from the other, and compare compute costs.

2.1 Models

We pre-train ten models (Table 1): one from-scratch encoder, five from-scratch seq2seq models, one encoder from a seq2seq model with continued MLM training, and three two-stage seq2seq models warm-started with the from-scratch encoder. We report the pre-training Compute Cost for each, where "TU" (Training Units) is defined as 100k update steps for 12 model layers with hidden dimension 1024 and batch size 1M tokens (Appendix D, E).

2.1.1 Encoder Model From Scratch

We train an encoder model ("roberta-12e" in Table 1) following a similar recipe to XLM-RoBERTa (Conneau et al., 2020a), using the MLM objectve (Figure 2a) of randomly masking 15% of subword tokens, as introduced in BERT (Devlin et al., 2019). We use a batch size of 1M tokens and train for 500k update steps. Notably, these settings match our seq2seq models. We use "PreLayerNorm" (Xiong et al., 2020), moving the layer norms to inside residual blocks to improve training stability.

2.1.2 Seq2Seq Objectives

Our seq2seq training follows the architecture and de-noising task of BART and mBART (Lewis et al., 2020; Liu et al., 2020); the only architecture change we make is to again use PreLayerNorm.

The de-noising objective selects 15% of the tokens in the input (spans of length $\sim$ Poisson(3)), and either (i) simply drops them, or (ii) replaces each selected span with a single mask token. The model is trained to reconstruct the original input entirely. See Figures 2b and 2c, respectively. We add a suffix "-mask" to the model names that use masking instead of dropping the tokens. Intuitively, adding an explicit mask token for de-noising makes the reconstruction task easier, as the decoder knows exactly where the missing tokens are needed.

2.1.3 Seq2Seq Models From Scratch

All of our seq2seq models use 12 encoder layers ("12e"). The first five models are trained from scratch starting from randomly initialized weights. The models "bart-12e12d" and "bart-12e12d-mask" use 12-layer decoders (same number as encoder layers) using the seq2seq de-noising training objective without masking and with masking, respectively. The remaining three models use a smaller decoder of either 2 layers ("bart-12e2d" without masking, "bart-12e2d-mask" with masking) or 1 layer ("bart-12e1d-mask", with masking). We hypothesize that reducing the size of the decoder may strengthen the encoder when it is extracted and used on its own.

2.1.4 Recipe 1: Encoder of Seq2Seq + MLM

We extract the encoder from the seq2seq model "bart-12e12d" and continue training via MLM for 100k updates ("bart-12e12d+mlm"). We initialize the MLM head from the input embedding and untie.

2.1.5 Recipe 2: Two-Stage Seq2Seq Models

Finally, we train three seq2seq models following the two-stage setup (Figure 1). We initialize the encoder weights of the seq2seq model with the MLM encoder "roberta-12e" (Section 2.1.1) and train via seq2seq de-noising without masking. The first two models train for 500k updates with the encoder always frozen: "2stage-bart-12e12d" uses standard cross-attention, where the decoder attends to only the final encoder layer, and "2stage-bart-12e12d-attn-f" uses a novel application of **attention fusion** (Cao et al., 2022) during cross-attention, where the decoder attends to all encoder layers.

Model	Encoder Layers	Decoder Layers	Encoder Updates	Decoder Updates	Compute Cost (TU)
Encoder Model From Scratch (MLM only)					
roberta-12e	12	0	500k	0	5.0
Seq2Seq Models From Scratch (de-noising only)					
bart-12e12d	12	12	500k	500k	10.0
bart-12e12d-mask	12	12	500k	500k	10.0
bart-12e2d	12	2	500k	500k	5.8
bart-12e2d-mask	12	2	500k	500k	5.8
bart-12e1d-mask	12	1	500k	500k	5.4
Recipe 1: Encoder of Seq2Seq + MLM					
bart-12e12d+mlm	12	12	500k (s2s) + 100k	500k	10.0 (s2s) + 1.0 = 11.0
Recipe 2: Two-Stage Seq2Seq Models (warm-start with MLM encoder)					
2stage-bart-12e12d	12	12	500k (MLM)	500k	5.0 (MLM) + 7.5 = 12.5
2stage-bart-12e12d-attn-f	12	12	500k (MLM)	500k	5.0 (MLM) + 7.5 = 12.5
2stage-bart-12e12d-unfrz	12	12	500k (MLM) + 150k	200k + 150k	5.0 (MLM) + 6.0 = 11.0

Table 1: Model architecture details. All models use a batch size of 1M tokens with hidden dimension of 1024, feed-forward dimension of 4096 and 16 attention heads.

The last model, "2stage-bart-12e12d-unfrz" uses standard cross-attention and **unfreezes the encoder partway through training**, applying 200k update steps with the encoder frozen, then 150k update steps with the encoder unfrozen.

In all cases, we initialize and tie the decoder embeddings from/to the encoder embeddings and keep them frozen as long as the encoder is frozen. The LM head is also initialized from the encoder embeddings, but it is untied from the embeddings and unfrozen from the beginning of the training.

2.1.6 Compute Cost Comparison

The baseline of training both models from scratch has a compute cost of 15.0 TU: 5.0 TU for "roberta-12e" plus 10.0 TU for "bart-12e12d". Our proposed recipes reduce the total compute cost either by 17% (to 12.5 TU) or by 27% (to 11.0 TU).

2.2 Pretraining Dataset

We pre-train on a combination of Wikipedia and mC4 (Xue et al., 2021) data in 12 languages: Arabic, English, French, German, Hindi, Italian, Japanese, Marathi, Portuguese, Spanish, Tamil, and Telugu. We pack sequences of tokens to produce sequences of approximately 512 subword units. We allow unrelated content to be packed together in the same sequence, separated with a special symbol "[DOC]". Maintaining a relatively constant number of subword sequences reduces padding and results in efficient compute. We up-sample data for different languages following Conneau et al. (2020a).

3 Fine-Tuning Results

We present the results on fine-tuning our pre-trained models. All runs are averaged over three random seeds and reported as mean ± standard deviation. See Appendix C for hyperparameters.

3.1 Encoder Model Results

In Table 2, we compare the performance of our encoder models on four datasets: (1) XNLI (Conneau et al., 2018) sentence-pair classification, (2) mATIS++ (Xu et al., 2020) joint Intent Classification (IC) and Slot Labeling (SL), (3) WikiANN (Pan et al., 2017) token-level Named Entity Recognition (NER), and (4) UDPOS (Nivre et al., 2020) token-level Part-of-Speech tagging (POS) (XTREME (Hu et al., 2020) version). For each task, we follow the cross-lingual zero-shot setting: train and validate on English data only, then report on the test set in English ("en") and the average over the zero-shot langauges ("avg-0s"). Appendix B shows results on each language.

We find that the MLM encoder performs best on all tasks except for mATIS++ IC avg-0s setting. The encoder of seq2seq ("bart-12e12d") is only slightly behind on the sentence-level tasks, on en/avg-0s by 0.6/1.1 points on XNLI (83.9 vs. 84.5 / 74.7 vs. 75.8), and 1.0/1.0 points on mATIS++ IC (96.8 vs. 97.8 / 86.2 vs. 87.2). However, **the gap is much larger on the sequence labeling tasks**: on en/avg-0s, 3.2/17.3 points on mATIS++ SL (92.5 vs. 95.7 / 44.3 vs. 61.6), 6.4/9.0 points on WikiANN NER (76.6 vs. 83.0 / 52.1 vs. 61.1), and

Encoder	Classification				Sequence Labeling					
	XNLI (acc.)		mATIS++ IC (acc.)		mATIS++ SL (f1)		WikiANN (f1)		UDPOS (f1)	
	en	avg-0s	en	avg-0s	en	avg-0s	en	avg-0s	en	avg-0s
Encoder Model From Scratch (MLM only)										
roberta-12e	**84.5**$_{\pm 0.5}$	**75.8**$_{\pm 0.2}$	**97.8**$_{\pm 0.1}$	87.2$_{\pm 4.1}$	**95.7**$_{\pm 0.1}$	**61.6**$_{\pm 0.6}$	**83.0**$_{\pm 0.1}$	**61.1**$_{\pm 0.4}$	**95.8**$_{\pm 0.0}$	**73.5**$_{\pm 0.2}$
Encoder of Seq2Seq Models (de-noising only)										
bart-12e12d	<u>83.9</u>$_{\pm 0.2}$	74.7$_{\pm 0.3}$	96.8$_{\pm 0.1}$	86.2$_{\pm 1.5}$	92.5$_{\pm 0.3}$	44.3$_{\pm 1.3}$	76.6$_{\pm 0.2}$	52.1$_{\pm 0.9}$	94.3$_{\pm 0.7}$	<u>61.5</u>$_{\pm 0.4}$
bart-12e12d-mask	<u>83.9</u>$_{\pm 0.4}$	<u>75.0</u>$_{\pm 0.6}$	97.1$_{\pm 0.1}$	87.3$_{\pm 0.7}$	91.1$_{\pm 0.9}$	41.3$_{\pm 1.3}$	73.2$_{\pm 0.1}$	48.4$_{\pm 0.6}$	93.3$_{\pm 0.1}$	55.1$_{\pm 0.4}$
bart-12e2d	71.3$_{\pm 0.1}$	59.7$_{\pm 0.5}$	96.1$_{\pm 0.1}$	79.1$_{\pm 0.8}$	91.4$_{\pm 0.1}$	38.2$_{\pm 1.7}$	69.3$_{\pm 0.5}$	42.9$_{\pm 0.1}$	92.1$_{\pm 0.1}$	50.7$_{\pm 0.5}$
bart-12e2d-mask	82.9$_{\pm 0.3}$	73.8$_{\pm 0.2}$	96.8$_{\pm 0.1}$	**88.1**$_{\pm 0.9}$	92.3$_{\pm 0.3}$	48.0$_{\pm 1.4}$	76.5$_{\pm 0.2}$	<u>54.0</u>$_{\pm 0.6}$	93.3$_{\pm 0.1}$	54.0$_{\pm 0.6}$
bart-12e1d-mask	82.4$_{\pm 0.2}$	72.7$_{\pm 0.1}$	97.0$_{\pm 0.1}$	<u>87.6</u>$_{\pm 0.5}$	92.8$_{\pm 0.5}$	49.3$_{\pm 1.2}$	74.6$_{\pm 0.5}$	48.5$_{\pm 0.3}$	92.4$_{\pm 0.1}$	46.3$_{\pm 1.7}$
Recipe 1: Encoder of Seq2Seq Model + MLM										
bart-12e12d+mlm	80.3$_{\pm 0.4}$	69.0$_{\pm 0.4}$	<u>97.2</u>$_{\pm 0.4}$	83.9$_{\pm 1.6}$	<u>95.3</u>$_{\pm 0.2}$	<u>56.5</u>$_{\pm 2.8}$	<u>79.9</u>$_{\pm 0.2}$	47.5$_{\pm 0.5}$	<u>95.1</u>$_{\pm 0.0}$	60.7$_{\pm 0.9}$

Table 2: Encoder results per task, English and avg. zero-shot. The best (second) mean result is bolded (underlined).

Seq2Seq Models	mTOP (acc.)		XSUM (ROUGE)		
	en	avg-0s	R-1	R-2	R-L
Seq2Seq Models From Scratch (de-noising only)					
bart-12e12d	**83.4**$_{\pm 0.2}$	45.7$_{\pm 1.1}$	<u>40.37</u>$_{\pm 0.07}$	17.37$_{\pm 0.06}$	32.46$_{\pm 0.06}$
bart-12e12d-mask	83.2$_{\pm 0.5}$	<u>46.9</u>$_{\pm 0.5}$	**40.63**$_{\pm 0.09}$	<u>17.48</u>$_{\pm 0.10}$	<u>32.63</u>$_{\pm 0.06}$
Recipe 2: Two-Stage Seq2Seq Models (warm-start with MLM encoder)					
2stage-bart-12e12d	82.0$_{\pm 1.1}$	46.8$_{\pm 1.1}$	40.12$_{\pm 0.06}$	17.13$_{\pm 0.03}$	32.16$_{\pm 0.01}$
2stage-bart-12e12d-attn-f	80.6$_{\pm 1.3}$	46.4$_{\pm 0.5}$	40.13$_{\pm 0.06}$	17.24$_{\pm 0.07}$	32.28$_{\pm 0.03}$
2stage-bart-12e12d-unfrz	<u>83.3</u>$_{\pm 0.2}$	**48.2**$_{\pm 0.5}$	**40.63**$_{\pm 0.11}$	**17.58**$_{\pm 0.03}$	**32.65**$_{\pm 0.05}$

Table 3: Seq2Seq results on mTOP cross-lingual semantic parsing and XSUM English summarization.

1.5/12.0 on UDPOS (94.3 vs. 95.8 / 61.5 vs. 73.5). This suggests that seq2seq pre-training may give the encoder the knowledge to perform sentence-level tasks, while MLM pre-training may be particularly effective for sequence labeling tasks which use the token-level representations directly.

With a 12-layer decoder, the explicit mask token during seq2seq pre-training does not seem to improve the encoder. However, when the decoder has only 2 layers, the mask token is crucial: "bart-12e2d-mask" out-performs "bart-12e2d" by a wide margin across tasks. We hypothesize that the mask token makes de-noising easier, by signaling where tokens should be filled in, and without this signal, the task is too challenging for a seq2seq model with just a 2-layer decoder. Reducing the decoder further to only 1 layer does not benefit the encoder.

Continuing training the seq2seq-extracted encoder on MLM for 100k updates does not close the gap to the from-scratch encoder across datasets. Some tasks improve, while others degrade.

3.2 Seq2Seq Model Results

We evaluate the generation quality of our seq2seq models on two datasets: mTOP (Li et al., 2021) cross-lingual zero-shot semantic parsing, and XSUM (Narayan et al., 2018) English summarization. For mTOP, following CLASP (Rosenbaum et al., 2022a), we use space-joined tokens as input, word sentinels, and SCIEM (Space- and Case-Insensitive Exact Match) metric. For both datasets, we generate outputs using beam search with k=3.

As shown in Table 3, **the two-stage model with encoder unfrozen partway through training is on-par with the from-scratch seq2seq model**: compared to "bart-12e12d ", "2stage-bart-12e12d-unfrz" is only 0.1 points behind on mTOP en (83.3 vs. 83.4) yet 2.5 points ahead on cross-lingual zero-shot (48.2 vs. 45.7). On XSUM, the two-stage model is on-par or slightly better than the from-scratch seq2seq models.

Masking during seq2seq pre-training does not greatly impact generation quality. When the encoder is frozen ("2stage-bart-12e12d"), the results are slightly behind; attention fusion ("2stage-bart-12e12d-attn-f") does not provide a clear benefit.

Overall, our proposed **two-stage seq2seq pre-training recipe provides both a multilingual encoder and a seq2seq model on-par with the two models trained from scratch, while reducing compute cost by 27%** (from 15.0 to 11.0 TU).

4 Conclusion and Future Work

In this work, we studied recipes to efficiently pre-train both a multilingual encoder and a seq2seq model by re-using the weights from one model for the other. We found that the most effective recipe is to start training of a seq2seq model from a pre-trained encoder and unfreeze it partway through the training. Future work can explore even more efficient pre-training strategies such as jointly training on MLM and sequence-level de-noising objectives, and probe further why the encoders trained as part of a seq2seq model do not do well on sequence labeling tasks.

5 Limitations

Our proposed two-stage training recipe is beneficial under the assumption that a pre-trained model is needed for generative as well as sequence labeling tasks. We believe that is typically the case, as one tries to offset the pre-training investment by using the model for as many tasks as possible, but this assumption might not apply in all cases. While we assess the effect of randomness on fine-tuning results by using multiple seeds, we have not done that for the pre-training itself. Even at our medium-size scale, it is already prohibitively expensive to do so. The evidence for the effectiveness of the two-stage approach is also limited by the number of tasks evaluated (2 sequence classification tasks, 2 sequence labeling tasks, 2 generation tasks), but we believe it is a reasonable trade-off between robust results and compute investment.

6 Acknowledgments

We thank Kai-Wei Chang, Nicolas Guenon Des Mesnards, and the anonymous ACL reviewers for their helpful feedback on our work.

References

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. *ArXiv*, abs/2002.12804.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Jin Cao, Chandana Satya Prakash, and Wael Hamza. 2022. Attention fusion: a light yet efficient late fusion mechanism for task adaptation in NLU. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 857–866, Seattle, United States. Association for Computational Linguistics.

Cheng Chen, Yichun Yin, Lifeng Shang, Xin Jiang, Yujia Qin, Fengyu Wang, Zhi Wang, Xiao Chen, Zhiyuan Liu, and Qun Liu. 2022. bert2BERT: Towards reusable pretrained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2134–2148, Dublin, Ireland. Association for Computational Linguistics.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Baindoor Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Oliveira Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. PaLM: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020a. Unsupervised cross-lingual representation learning at scale. In *ACL*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020b. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, M. Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *ArXiv*, abs/1905.03197.

Jack G. M. FitzGerald, Shankar Ananthakrishnan, Konstantine Arkoudas, David Bernardi, Abhishek Bhagia, Claudio Delli Bovi, Jin Cao, Rakesh Chada, Amit Chauhan, Luoxin Chen, Anurag Dwarakanath, Satyam Dwivedi, Turan Gojayev, Karthik Gopalakrishnan, Thomas Gueudré, Dilek Z. Hakkani-Tür, Wael Hamza, Jonathan Hueser, Kevin Martin Jose, Haidar Khan, Beiye Liu, Jianhua Lu, A. Manzotti, Pradeep Natarajan, Karolina Owczarzak, Gokmen Oz, Enrico Palumbo, Charith S. Peris, Chandan Prakash, Stephen Rawls, Andrew Rosenbaum, Anjali Shenoy, Saleh Soltan, Mukund Sridhar, Liz Tan, Fabian Triefenbach, Pang Wei, Haiyang Yu, Shuai Zheng, Gokhan Tur, and Premkumar Natarajan. 2022. Alexa teacher model: Pretraining and distilling multi-billion-parameter encoders for natural language understanding systems.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online. Association for Computational Linguistics.

Frederick Liu, Terry Huang, Shihang Lyu, Siamak Shakeri, Hongkun Yu, and Jing Li. 2022. Enct5: A framework for fine-tuning t5 as non-autoregressive models.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Hiroki Nakayama. 2018. seqeval: A python framework for sequence labeling evaluation. Software available from https://github.com/chakki-works/seqeval.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pretrained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajivc, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis M. Tyers, and Daniel Zeman. 2020. Universal dependencies v2: An evergrowing multilingual treebank collection. In *International Conference on Language Resources and Evaluation*.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*

Papers), pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020b. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '20, page 3505–3506, New York, NY, USA. Association for Computing Machinery.

Andy Rosenbaum, Saleh Soltan, Wael Hamza, Marco Damonte, Isabel Groves, and Amir Saffari. 2022a. CLASP: Few-shot cross-lingual data augmentation for semantic parsing. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 444–462, Online only. Association for Computational Linguistics.

Andy Rosenbaum, Saleh Soltan, Wael Hamza, Yannick Versley, and Markus Boese. 2022b. LINGUIST: Language model instruction tuning to generate annotated utterances for intent classification and slot tagging. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 218–241, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang A. Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M SAIFUL BARI, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Rose Biderman, Leo Gao, T. G. Owe Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask prompted training enables zero-shot task generalization. *ArXiv*, abs/2110.08207.

Saleh Soltan, Shankar Ananthakrishnan, Jack G. M. FitzGerald, Rahul Gupta, Wael Hamza, Haidar Khan, Charith S. Peris, Stephen Rawls, Andrew Rosenbaum, Anna Rumshisky, Chandan Prakash, Mukund Sridhar, Fabian Triefenbach, Apurv Verma, Gokhan Tur, and Premkumar Natarajan. 2022. Alexatm 20b: Few-shot learning using a large-scale multilingual seq2seq model. *ArXiv*, abs/2208.01448.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam M. Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, Yaguang Li, Hongrae Lee, Huaixiu Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Yanqi Zhou, Chung-Ching Chang, I. A. Krivokon, Willard James Rusch, Marc Pickett, Kathleen S. Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Hartz Søraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Díaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravindran Rajakumar, Alena Butryna, Matthew Lamm, V. O. Kuzmina, Joseph Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Aguera-Arcas, Claire Cui, Marian Croak, Ed Chi, and Quoc Le. 2022. Lamda: Language models for dialog applications. *ArXiv*, abs/2201.08239.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi 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 Rush. 2020. 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. Association for Computational Linguistics.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. 2020. On layer normalization in the transformer architecture. *ArXiv*, abs/2002.04745.

Weijia Xu, Batool Haider, and Saab Mansour. 2020. End-to-end slot alignment and recognition for cross-lingual NLU. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5052–5063, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. 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. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068.

A Additional Related Work

Pre-trained Transformer models (Vaswani et al., 2017) are commonly used in Natural Language Processing (NLP) for both transfer learning in downstream tasks (Devlin et al., 2019; Liu et al., 2019; Radford and Narasimhan, 2018; Radford et al., 2019) and for in-context learning (Brown et al., 2020). Transformers were originally designed as sequence-to-sequence (seq2seq) models with an encoder and a decoder component (Vaswani et al., 2017). However, all three obvious variants of this architecture are now common: encoder-only (Devlin et al., 2019), decoder-only (Radford and Narasimhan, 2018; Radford et al., 2019; Brown et al., 2020; Chowdhery et al., 2022; Zhang et al., 2022; Thoppilan et al., 2022) and seq2seq (Lewis et al., 2020; Raffel et al., 2020b; Sanh et al., 2021; Dong et al., 2019; Bao et al., 2020).

Commonly, encoder transformer models are pre-trained using the MLM objective (Devlin et al., 2019). Decoders are pre-trained using a next-token left-to-right prediction (causal) language modeling objective (Radford and Narasimhan, 2018) or some version of autoregressive de-noising (Lewis et al., 2020). Seq2seq models often combine these objectives (Lewis et al., 2020; Bao et al., 2020).

We follow the multilingual approach of models such as XLM-RoBERTa (Conneau et al., 2020b) (encoder-only) and mT5/mBART (Xue et al., 2021; Liu et al., 2020) (seq2seq), where the model is pre-trained on data from multiple languages. This enables **cross-lingual zero-shot fine-tuning**, where the model is fine-tuned on task data only from a single language (usually English), then evaluated on multiple languages.

Previous literature has explored using a pre-trained encoder to initialize a larger encoder (Chen et al., 2022) or a seq2seq model (Rothe et al., 2020). The latter was applied to large-scale models, e.g. AlexaTM 20B and AlexaTM 5B (Soltan et al., 2022; Rosenbaum et al., 2022b; FitzGerald et al., 2022). Our work provides the first direct comparison of warm-starting vs. from-scratch seq2seq pre-training using the same data and codebase.

Recently, Sentence-T5 (Ni et al., 2022) studied the opposite direction, showing that extracting the encoder from T5 (Raffel et al., 2020a) can out-perform BERT on several sentence-level tasks. We also explore extracting the encoder from a seq2seq model, adding the novelty of the first explicit comparison with MLM encoders using the same pre-training data and codebase. Furthermore, whereas Sentence-T5 studies only sentence level tasks in English, we study both sentence-level and token-level (e.g. sequence labeling) multilingual tasks. We show that the encoder extracted from a seq2seq model under-performs on token-level tasks, motivating our proposed sequential pre-training recipes.

EncT5 (Liu et al., 2022) proposes an alternative method to fine-tune the encoder from a seq2seq model for classification and sequence labeling tasks, by attaching a randomly initialized one-layer decoder with cross-attention. They report substantial improvements on UDPOS (Part-of-Speech tagging, a sequence labeling task) compared to an mBERT (MLM encoder) model of similar encoder size, however the comparison is between models pre-trained on different data and codebases. For a cleaner comparison, we would need to implement and evaluate the EncT5 framework with our models, which is challenging since no reference implementation is available, and also because Liu et al. (2022) provide only the average number across languages for UDPOS and do not report per language. Therefore, we defer a more thorough study of EncT5 vs. standard feed-forward layer classification heads to future work.

B Results by Language

Our main results in Section 3 (Tables 2 and 3) show only the English and average zero-shot results for brevity. Here, for completeness, we show the results on each langauge for XNLI (Table 4), mATIS++ Intent Classification (IC) (Table 5), mATIS++ Slot Labeling (SL) (Table 6), WikiANN NER (Table 7), UDPOS (Table 8), and mTOP semantic parsing (Table 9).

C Fine-tuning Hyperparameters

Table 10 shows the hyperparameters for fine-tuning the pre-trained models. For encoders, we first performed a single run with learning rates among 1e-6, 3e-6, 1e-5, 3e-5, 1e-4 for each task and model, and found that the best learning rate was nearly always 1e-5 or 3e-5, with only small differences between

Encoder	en	ar	de	es	fr	hi	avg-0s
Encoder Model From Scratch (MLM only)							
roberta-12e	**84.5**$_{\pm 0.5}$	**72.9**$_{\pm 0.2}$	**76.9**$_{\pm 0.3}$	**79.9**$_{\pm 0.2}$	**78.7**$_{\pm 0.3}$	**70.5**$_{\pm 0.8}$	**75.8**$_{\pm 0.2}$
Encoder of Seq2Seq Models (de-noising only)							
bart-12e12d	83.9$_{\pm 0.2}$	71.6$_{\pm 0.6}$	76.0$_{\pm 0.5}$	79.2$_{\pm 0.8}$	77.8$_{\pm 0.1}$	68.9$_{\pm 0.3}$	74.7$_{\pm 0.3}$
bart-12e12d-mask	83.9$_{\pm 0.4}$	71.9$_{\pm 0.7}$	76.3$_{\pm 0.2}$	79.5$_{\pm 0.6}$	78.5$_{\pm 0.5}$	68.5$_{\pm 1.3}$	75.0$_{\pm 0.6}$
bart-12e2d	71.3$_{\pm 0.1}$	56.7$_{\pm 0.7}$	60.3$_{\pm 0.3}$	64.2$_{\pm 0.7}$	63.9$_{\pm 0.6}$	53.3$_{\pm 0.2}$	59.7$_{\pm 0.5}$
bart-12e2d-mask	82.9$_{\pm 0.3}$	70.9$_{\pm 0.4}$	74.7$_{\pm 0.5}$	78.1$_{\pm 0.3}$	76.9$_{\pm 0.4}$	68.2$_{\pm 0.5}$	73.8$_{\pm 0.2}$
bart-12e1d-mask	82.4$_{\pm 0.2}$	69.6$_{\pm 0.3}$	73.5$_{\pm 0.3}$	77.0$_{\pm 0.1}$	76.3$_{\pm 0.4}$	66.9$_{\pm 0.2}$	72.7$_{\pm 0.1}$
Recipe 1: Encoder of Seq2Seq Model + MLM							
bart-12e12d+mlm	80.3$_{\pm 0.4}$	65.6$_{\pm 0.4}$	70.6$_{\pm 0.2}$	72.9$_{\pm 0.8}$	72.6$_{\pm 0.2}$	63.4$_{\pm 0.8}$	69.0$_{\pm 0.4}$

Table 4: Encoder model results by language on XNLI test sets: accuracy.

Encoder	en	de	es	fr	hi	ja	pt	avg-0s
Encoder Model From Scratch (MLM only)								
roberta-12e	**97.8**$_{\pm 0.1}$	**92.7**$_{\pm 2.2}$	**96.2**$_{\pm 0.5}$	94.6$_{\pm 1.4}$	**79.5**$_{\pm 4.5}$	65.6$_{\pm 17.1}$	94.3$_{\pm 2.4}$	87.2$_{\pm 4.1}$
Encoder of Seq2Seq Models (de-noising only)								
bart-12e12d	96.8$_{\pm 0.1}$	91.0$_{\pm 2.5}$	91.0$_{\pm 0.4}$	93.1$_{\pm 1.5}$	77.7$_{\pm 3.6}$	72.1$_{\pm 4.5}$	92.2$_{\pm 1.8}$	86.2$_{\pm 1.5}$
bart-12e12d-mask	97.1$_{\pm 0.1}$	89.7$_{\pm 1.1}$	94.2$_{\pm 0.4}$	94.0$_{\pm 0.8}$	78.6$_{\pm 0.7}$	75.0$_{\pm 3.4}$	91.9$_{\pm 1.1}$	87.3$_{\pm 0.7}$
bart-12e2d	96.1$_{\pm 0.1}$	80.4$_{\pm 8.1}$	84.7$_{\pm 3.5}$	86.1$_{\pm 3.0}$	74.3$_{\pm 1.3}$	64.4$_{\pm 5.6}$	84.6$_{\pm 2.6}$	79.1$_{\pm 0.8}$
bart-12e2d-mask	96.8$_{\pm 0.1}$	92.4$_{\pm 0.6}$	94.5$_{\pm 0.5}$	**94.7**$_{\pm 0.5}$	79.1$_{\pm 1.5}$	73.9$_{\pm 5.1}$	**94.4**$_{\pm 0.4}$	**88.1**$_{\pm 0.9}$
bart-12e1d-mask	97.0$_{\pm 0.1}$	90.1$_{\pm 0.8}$	94.8$_{\pm 0.4}$	93.3$_{\pm 0.4}$	79.4$_{\pm 0.8}$	**76.5**$_{\pm 3.0}$	91.6$_{\pm 0.4}$	87.6$_{\pm 0.5}$
Recipe 1: Encoder of Seq2Seq Model + MLM								
bart-12e12d+mlm	97.2$_{\pm 0.4}$	86.1$_{\pm 4.2}$	92.0$_{\pm 0.8}$	91.1$_{\pm 1.5}$	76.1$_{\pm 2.7}$	70.1$_{\pm 4.4}$	88.2$_{\pm 3.3}$	83.9$_{\pm 1.6}$

Table 5: Encoder model results by language on mATIS++ test sets, Intent Classificaiton (IC) accuracy.

those two options by model. For consistency, we then fixed the learning rate for each task and ran each model on each task with three random seeds. We use Adam (Kingma and Ba, 2015) optimization. We freeze the embedding layer which we find generally slightly improves the cross-lingual zero-shot results.

For XNLI, we follow the standard practice established in BERT (Devlin et al., 2019) to attach the classification head to the first token ("<s>" for all of our models). We also explored max pooling across all tokens and did not observe a significant difference in performance.

For mATIS++, following Chen et al. (2019), we use two separate classification heads, one for Intent Classification (IC) attached to the encoder output of the first subword token of the sequence, and the second for Slot Labeling (SL) attached to the *first subword of each whole word* in the sequence.

Similarly, for WikiANN NER and UDPOS, we again use a single classification head attached to the first subword of each whole word in the sequence. When computing f1 score for sequence labeling tasks (mATIS++ SL and WikiANN NER), we ignore the "O" ("Outside") tag, using the seqeval (Nakayama, 2018) implementation which takes into account the BIO tags present in WikiANN.

D Details on Compute Cost

We provide details on the compute cost reported in Table 1. The unit "TU" (Training Updates) is defined as the compute cost for 100k updates (forward and backward pass) of 12 model layers with hidden dimension 1024 and batch size 1M tokens. The encoder-only MLM model trains for 500k updates, for Compute Cost 5.0 TU. The Seq2Seq Models From Scratch have more layers, and therefore a larger Compute Cost for 500k updates. For example, "bart-12e12d" has 12 layers each for encoder and decoder, resulting in a compute cost of 10.0 for 500k updates. As a baseline, training both the MLM encoder and the seq2seq models from scratch would incur a compute cost of $5.0 + 10.0 = 15.0$ TU.

Encoder	en	de	es	fr	hi	ja	pt	avg-0s
Encoder Model From Scratch (MLM only)								
roberta-12e	**95.7**$_{\pm 0.1}$	**82.8**$_{\pm 1.2}$	**81.8**$_{\pm 0.6}$	**72.3**$_{\pm 1.7}$	31.8$_{\pm 1.3}$	**20.9**$_{\pm 1.2}$	**79.8**$_{\pm 0.8}$	**61.6**$_{\pm 0.6}$
Encoder of Seq2Seq Models (de-noising only)								
bart-12e12d	92.5$_{\pm 0.3}$	57.0$_{\pm 4.8}$	52.6$_{\pm 0.9}$	58.5$_{\pm 0.4}$	25.1$_{\pm 1.3}$	12.6$_{\pm 1.5}$	60.0$_{\pm 3.1}$	44.3$_{\pm 1.3}$
bart-12e12d-mask	91.1$_{\pm 0.9}$	54.7$_{\pm 2.6}$	52.9$_{\pm 2.4}$	52.5$_{\pm 0.9}$	23.2$_{\pm 2.9}$	10.5$_{\pm 1.5}$	53.9$_{\pm 1.7}$	41.3$_{\pm 1.3}$
bart-12e2d	91.4$_{\pm 0.1}$	52.0$_{\pm 4.1}$	53.2$_{\pm 2.3}$	50.3$_{\pm 1.7}$	11.4$_{\pm 0.8}$	3.9$_{\pm 1.5}$	58.3$_{\pm 2.2}$	38.2$_{\pm 1.7}$
bart-12e2d-mask	92.3$_{\pm 0.3}$	63.7$_{\pm 2.5}$	60.7$_{\pm 1.1}$	60.8$_{\pm 1.8}$	26.0$_{\pm 1.5}$	10.4$_{\pm 0.9}$	66.4$_{\pm 2.1}$	48.0$_{\pm 1.4}$
bart-12e1d-mask	92.8$_{\pm 0.5}$	65.6$_{\pm 4.8}$	59.5$_{\pm 0.4}$	61.7$_{\pm 0.3}$	24.7$_{\pm 1.7}$	16.3$_{\pm 1.8}$	68.0$_{\pm 1.6}$	49.3$_{\pm 1.2}$
Recipe 1: Encoder of Seq2Seq Model + MLM								
bart-12e12d+mlm	95.3$_{\pm 0.2}$	70.5$_{\pm 7.6}$	79.7$_{\pm 1.2}$	67.3$_{\pm 1.4}$	**33.5**$_{\pm 3.9}$	15.9$_{\pm 3.8}$	72.2$_{\pm 0.6}$	56.5$_{\pm 2.8}$

Table 6: Encoder model results by language on mATIS++ test sets, Slot Labeling (SL) f1 score.

Encoder	en	ar	de	es	fr	hi	it	ja	mr	pt	ta	te	avg-0s
Encoder Model From Scratch (MLM only)													
roberta-12e	**83.0** ±0.1	**45.7** ±2.1	**73.5** ±0.7	**68.9** ±0.5	**75.4** ±0.4	**71.5** ±0.5	**76.7** ±0.7	**28.0** ±1.0	**57.5** ±2.0	**74.7** ±0.2	**53.9** ±0.3	**46.5** ±0.7	**61.1** ±0.4
Encoder of Seq2Seq Models (de-noising only)													
bart-12e12d	76.6 ±0.2	44.4 ±1.8	64.8 ±1.2	61.1 ±1.7	70.7 ±0.8	62.2 ±2.2	69.7 ±0.7	10.2 ±0.6	41.0 ±1.9	69.7 ±0.4	42.0 ±0.2	37.0 ±2.3	52.1 ±0.9
bart-12e12d-mask	73.2 ±0.1	30.6 ±0.8	57.7 ±0.3	59.8 ±0.5	67.4 ±0.5	60.7 ±0.1	66.1 ±0.5	8.1 ±0.7	41.3 ±2.9	68.9 ±1.5	37.6 ±1.1	33.8 ±1.9	48.4 ±0.6
bart-12e2d	69.3 ±0.5	31.0 ±1.7	53.5 ±1.0	54.9 ±0.5	61.2 ±0.5	48.7 ±0.8	62.0 ±0.8	7.1 ±0.4	33.2 ±3.3	61.9 ±0.2	31.1 ±0.8	28.0 ±1.0	42.9 ±0.1
bart-12e2d-mask	76.5 ±0.2	45.2 ±1.9	63.5 ±0.5	65.0 ±2.4	70.8 ±0.5	64.0 ±1.0	69.9 ±0.3	10.3 ±0.7	44.4 ±3.3	71.5 ±0.5	47.7 ±2.4	41.4 ±3.1	54.0 ±0.6
bart-12e1d-mask	74.6 ±0.5	40.9 ±2.5	54.5 ±1.6	64.0 ±2.0	64.3 ±0.6	54.3 ±1.0	65.4 ±0.3	9.4 ±0.8	43.0 ±0.4	67.3 ±0.5	37.8 ±0.9	32.1 ±2.3	48.5 ±0.3
Recipe 1: Encoder of Seq2Seq Model + MLM													
bart-12e12d+mlm	79.9 ±0.2	29.8 ±1.0	62.8 ±0.6	60.9 ±0.5	68.9 ±0.3	58.7 ±1.4	68.7 ±0.4	13.6 ±0.3	29.4 ±0.8	69.5 ±0.7	33.7 ±1.1	27.0 ±0.9	47.5 ±0.5

Table 7: Encoder model results by language on WikiANN Named Entity Recognition (NER) test sets, f1 score.

Recipe 1 (Encoder of Seq2Seq + MLM), first pays compute cost 10.0 TU for the seq2seq training, then 1.0 TU for 100k MLM updates on the extracted encoder, for a total of 11.0 TU.

For Recipe 2 (Two-Stage Seq2Seq Models warm-started with MLM encoder), we first pay a compute cost of 5.0 TU from MLM pre-training of the encoder, then add compute cost for the second stage seq2seq pre-training. When the encoder is frozen, we only need to compute the forward pass for the encoder, not the backward pass. We estimate that when the encoder is frozen, its forward pass uses $1/2$ the compute as a forward and backward pass would use. (In reality, the ratio is likely less, as we also save memory by not needing to store the optimizer states.) Therefore, when the encoder is frozen for "2stage-bart-12e12d" and "2stage-bart-12e12d-attn-f", the 500k decoder updates incur a compute cost of 2.5 on the encoder side and 5.0 on the decoder side. Adding this to the 5.0 for MLM initialization gives a total compute cost of $5.0 + 7.5 + 12.5$ TU.

For "2stage-bart-12e12d-unfrz", the 200k updates with frozen encoder incur a compute cost of 1.0 TU on the encoder side and 2.0 TU on the decoder size for a total of 3.0 TU. During the final 150k updates, the encoder is unfrozen, so the compute cost is 3.0. Adding the 5.0 compute cost for MLM Encoder initialization, the total compute cost for this model is $5.0 + 3.0 + 3.0 = 11.0$ TU.

Encoder	en	ar	de	es	fr	hi	it	ja	mr	pt	ta	te	avg-0s
Encoder Model From Scratch (MLM only)													
roberta-12e	**95.8** ±0.0	**65.9** ±0.2	**86.2** ±0.2	**85.6** ±0.5	**77.6** ±0.2	**67.5** ±0.6	**88.0** ±0.4	**44.9** ±1.3	**74.4** ±1.7	**86.1** ±0.3	**55.1** ±0.1	**76.8** ±1.2	**73.5** ±0.2
Encoder of Seq2Seq Models (de-noising only)													
bart-12e12d	94.3 ±0.7	54.0 ±0.4	75.0 ±3.9	70.7 ±5.5	66.7 ±1.1	57.0 ±1.4	71.2 ±2.4	32.1 ±7.9	61.5 ±9.7	72.8 ±3.0	47.4 ±4.5	68.6 ±7.8	61.5 ±0.4
bart-12e12d-mask	93.3 ±0.1	49.0 ±1.3	62.4 ±1.2	58.9 ±0.5	56.4 ±0.4	50.6 ±1.4	60.9 ±0.6	27.3 ±1.1	60.0 ±1.1	64.3 ±0.5	47.3 ±0.5	69.0 ±0.9	55.1 ±0.4
bart-12e2d	92.1 ±0.1	43.5 ±0.9	60.8 ±1.5	58.4 ±1.7	54.6 ±1.4	42.3 ±0.3	58.5 ±1.3	16.8 ±0.3	57.2 ±3.0	63.2 ±0.9	41.4 ±0.3	61.1 ±1.2	50.7 ±0.5
bart-12e2d-mask	93.3 ±0.1	48.9 ±0.6	61.7 ±2.6	52.8 ±0.9	52.9 ±1.8	48.8 ±0.5	58.1 ±1.2	27.1 ±1.2	63.3 ±1.5	59.3 ±1.6	48.1 ±1.0	73.4 ±2.2	54.0 ±0.6
bart-12e1d-mask	92.4 ±0.1	44.8 ±1.4	52.5 ±3.4	43.8 ±1.6	43.8 ±2.8	43.0 ±2.6	47.9 ±3.2	19.8 ±1.7	58.4 ±1.9	53.0 ±1.5	42.1 ±1.1	60.0 ±1.9	46.3 ±1.7
Recipe 1: Encoder of Seq2Seq Model + MLM													
bart-12e12d+mlm	95.1 ±0.0	53.5 ±1.3	78.2 ±1.1	76.1 ±0.7	68.2 ±1.8	56.0 ±2.0	72.8 ±0.7	39.6 ±1.4	49.4 ±1.0	74.9 ±1.0	41.9 ±0.3	57.5 ±1.9	60.7 ±0.9

Table 8: Encoder model results by language on UDPOS Part-of-Speech tagging (POS) test sets, f1 score.

Model	en	fr	de	es	hi	avg-0s
Seq2Seq Models From Scratch (de-noising only)						
bart-12e12d	**83.4** ±0.2	54.3 ±1.2	48.5 ±1.7	51.6 ±1.6	28.4 ±0.5	45.7 ±1.1
bart-12e12d-mask	83.2 ±0.5	53.9 ±0.6	51.0 ±0.4	53.2 ±0.9	29.3 ±0.2	46.9 ±0.5
Recipe 2: Two-Stage Seq2Seq Models (warm-start with MLM encoder)						
2stage-bart-12e12d	82.0 ±1.1	52.3 ±.06	49.6 ±1.4	54.4 ±0.5	28.8 ±0.3	46.3 ±0.3
2stage-bart-12e12d-attn-f	80.6 ±1.3	52.6 ±0.7	49.8 ±0.7	53.7 ±0.8	29.7 ±0.4	46.4 ±0.5
2stage-bart-12e12d-unfrz	83.3 ±0.2	**55.2** ±1.1	**51.3** ±1.6	**55.3** ±1.2	**31.1** ±0.2	**48.2** ±0.5

Table 9: Seq2Seq model results by language on mTOP semantic parsing test sets, SCIEM.

E Pre-Training Details

We show an example sentence for each of our pre-training objectives in Figure 2.

Models were pre-trained (8 or 16 machines) and fine-tuned (1 machine) on AWS p3dn.24xlarge instances. For MLM pre-training, we use a peak learning rate of 1.5e-4 (1e-4 for the second stage of Recipe 1) warmed up over 5k update steps (1k for the second stage of Recipe 1) and decayed linearly down to 5e-6 over the total number of updates (500k or 100k, respectively). For seq2seq pre-training, we use the same learning rate as MLM pre-training: peak of 1.5e-4, warmed up over 5k updates, and linearly decayed down to 5e-6 for the duration of pre-training. For all pre-training runs, we use dropout of 0.1.

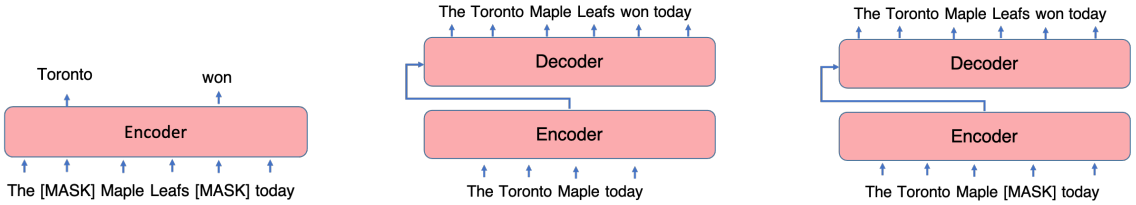
Our code is derived from HuggingFace (Wolf et al., 2020). We use DeepSpeed (Rasley et al., 2020) ZeRO Stage 1 to accelerate training.

F Dataset Sources

We show in Table 11 the source locations of the datasets we use for fine-tuning evaluation.

Parameter	Encoder tasks				Seq2Seq tasks	
	XNLI	mATIS++	WikiANN	UDPOS	mTOP	XSUM
Peak Learning Rate (LR)	1e-5	3e-5	3e-5	3e-5	5e-6	5e-6
LR warmup type	linear from 0	linear from 0	linear from 0	linear from 0	exponential from 1e-7	exponential from 1e-7
LR warmup num steps	1000	500	300	1000	1000	1000
LR decay type	linear to 0	linear to 0	linear to 0	linear to 0	linear to 1e-7	linear to 1e-7
Batch size	128	128	128	128	32	32
Epochs	5	200	20	56	200	200
Validation Metric	Accuracy	Slot Labeling f1	Slot Labeling f1	Slot Labeling f1	Exact Match	Perplexity
Max number of updates	30k	7k	3k	9k	∼50k	∼50k
Classification head(s)	[512] gelu	[256,256] gelu each for IC and SL	[512] gelu	[512] gelu	–	–

Table 10: Hyperparameters for fine-tuning. All models use AdamW with betas (0.9, 0.99), weight decay 0.1, and dropout 0.1. For each run, we select the checkpoint with the best value of the target metric on the validation set.



(a) Masked Language Modeling (MLM). (b) Seq2Seq de-noising. (c) Seq2Seq de-noising with masking.

Figure 2: The training objectives we explore, with the example sentence "The Toronto Maple Leafs won today".

Dataset	Source
XNLI	https://huggingface.co/datasets/xnli
mATIS++	https://github.com/amazon-science/multiatis
WikiANN	https://huggingface.co/datasets/wikiann
UDPOS	https://huggingface.co/datasets/xtreme
mTOP	https://fb.me/mtop_dataset
XSUM	https://huggingface.co/datasets/xsum

Table 11: Source Locations for the fine-tuning datasets we evaluate on.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Left blank.

☐ A2. Did you discuss any potential risks of your work?
Not applicable. ☐

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Not applicable. Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.