happens in the presence of gradient clipping [15]. To mitigate this problem, many approaches suggest restarting training from an earlier checkpoint [15, 33, 91], skipping 200-500 earlier data batches at the point of divergence in [15] and re-shuffling batches in [91]. The embedding layer gradient shrink proves to further stabilize the training as its gradient norm is significantly larger than the other layers [33]. Another suggestion to improve training stability for larger models is not to use **biases** in dense and norm layers as in [15].

Weight Initialization: It plays a significant role in model convergence and training stability. GPT-NeoX [118] initializes feed-forward layers before residuals with $\frac{2}{L\sqrt{d}}$ as in [153] and other layers with the small initialization scheme [298]. This avoids activations growing exponentially with increasing depth. MT-NLG [117] found higher variance for weight initialization leads to unstable training, hence validating small initialization scheme [298]. Various models perform random weight initialization which can cause bad initialization, Galactica [148] suggests a longer warmup to negate the effect.

Learning Rate: A suitable learning rate is important for stable training. It is suggested to use a lower value [13, 15, 124] with warmup and decay (cosine or linear). Usually, the learning rate is within the range $1e^{-4}$ to $8e^{-4}$. Moreover, MT-NLG (530B) [117] and GPT-NeoX (20B) [118] suggest interpolating learning rates based on the model size using the GPT-3 [6] models ranging between 13B and 175B. This avoids tuning the learning rate hyperparameter.

Training Parallelism: 3D parallelism, a combination of data, pipeline, and tensor parallelism, is the most utilized training parallelism approach in LLMs [33, 15, 14, 13, 117, 115, 112]. In addition to 3D parallelism, BLOOM [13] uses a zero optimizer [37] to shard optimizer states. PanGu-α [108] and PanGu-Σ [92] go beyond 3D parallelism and apply 5D parallelism which additionally contains optimizer parallelism and rematerialization.

Mode Switching: It adds task-related tokens at the beginning of the text during training. These tokens refer to the natural language understanding and natural language generation tasks which are shown to improve downstream task performance in [125, 124, 122]. During fine-tuning and inference, tokens are appended based on the downstream tasks.

Controllable Text Generation: Generating credible and controlled text from a pre-trained model is challenging. GPT-3 [6] and other LLMs use in-context learning to control generated text. While in-context learning helps in controlling the generated text, ERNIE 3.0 Titan [35] suggests using adversarial loss to rank its generated text for credibility and soft prompts such as genre, topic, keywords, sentiment, and length for better control on generated text.

3.8.3. Supervised Models vs Generalized Models

Although generalized models are capable of performing diverse tasks with good performance they have not yet outperformed models trained in supervised settings. The supervised trained models are still state-of-the-art in various NLP tasks by a large margin as shown in [6, 15, 18].

3.8.4. Zero-Shot vs Few-Shot

LLMs perform well in zero-shot and few-shot settings. But the performance difference between zero-shot and few-shot is large for pre-trained models [6, 15], naming LLMs as meta-learners [6]. LLMs zero-shot evaluations underperform unsupervised methods in neural machine translation [6]. The literature shows pre-training is not enough for good zero-shot performance [15, 16]. To improve the zero-shot performance the literature suggests using instruction fine-tuning that improves the zero-shot performance significantly and outperforms baselines. Instruction fine-tuning has also been shown to improve zero-shot generalization to unseen tasks. Another model, Flan-PaLM [16], unlocks zero-shot reasoning with CoT training.

3.8.5. Encoder vs Decoder vs Encoder-Decoder

Traditionally, these architectures perform well for different tasks, for example, encoder-only for NLU tasks, decoder-only for NLG, and encoder-decoder for sequence2sequence modeling. Encoder-only models are famous for smaller models such as Bert [7], RoBERTa [299], etc., whereas LLMs are either decoder-only [6, 118, 13] or encoder-decoder [10, 11, 122]. While decoder-only models are good at NLG tasks, various LLMs, PaLM [15], OPT [14], GPT-3 [6], BLOOM [13], LLaMA [156], are decoder-only models with significant performance gains on both NLU and NLG tasks. In contradiction to this, T5 [10] and UL2 [125] identify encoder-decoder models out-performing decoder-only models. In another study, PaLM [15] finds increasing the size of decoder-only models can reduce the performance gap between decoder-only and encoder-decoder architectures.

Although decoder-only architectures have become a trend for LLMs, many recently proposed approaches [125, 122] use mode-switching tokens in text with encoder-decoder architectures to enable task-specific modes. Similarly, CodeT5+ [34] uses an encoder-decoder architecture with multiple training objectives for different tasks, activating the encoder, decoder, or both according to the tasks. These variations in architecture and training objectives allow a model to perform well in different settings. Because of this dynamic configuration, the future of LLMs can be attributed to encoder-decoder architectures.

4. Model Configurations

We provide different statistics of pre-trained and instruction-tuned models in this section. This includes information such as publication venue, license type, model creators, steps trained, parallelism, etc in Table 3 and Table 4. Architecture details of pre-trained LLMs are available in Table 5. Providing these details for instruction-tuned models is unnecessary because it fine-tunes pre-trained models for instruction datasets. Hence, architectural details are the same as the baselines. Moreover, optimization settings for various LLMs are available in Table 6 and Table 7. We do not include details on precision, warmup, and weight decay in Table 7. These details are not as important as others to mention for instruction-tuned models, and are not provided by the papers.