But the field didn't stop there. Let's now explore some of the recent advances.

An Avalanche of Transformer Models

The year 2018 has been called the "ImageNet moment for NLP". Since then, progress has been astounding, with larger and larger transformer-based architectures trained on immense datasets.

First, the GPT paper[27] by Alec Radford and other OpenAI researchers once again demonstrated the effectiveness of unsupervised pretraining, like the ELMo and ULMFiT papers before it, but this time using a transformer-like architecture. The authors pretrained a large but fairly simple architecture composed of a stack of 12 transformer modules using only masked multi-head attention layers, like in the original transformer's decoder. They trained it on a very large dataset, using the same autoregressive technique we used for our Shakespearean char-RNN: just predict the next token. This is a form of self-supervised learning. Then they fine-tuned it on various language tasks, using only minor adaptations for each task. The tasks were quite diverse: they included text classification, *entailment* (whether sentence A imposes, involves, or implies sentence B as a necessary consequence),[28] similarity (e.g., "Nice weather today" is very similar to "It is sunny"), and question answering (given a few paragraphs of text giving some context, the model must answer some multiple-choice questions).

Then Google's BERT paper[29] came out: it also demonstrated the effectiveness of self-supervised pretraining on a large corpus, using a similar architecture to GPT but with nonmasked multi-head attention layers only, like in the original transformer's encoder. This means that the model is naturally bidirectional; hence the B in BERT (*Bidirectional Encoder Representations from Transformers*). Most importantly, the authors proposed two pretraining tasks that explain most of the model's strength:

Masked language model (MLM)

Each word in a sentence has a 15% probability of being masked, and the model is trained to predict the masked words. For example, if the original sentence is "She had fun at the birthday party", then the model may be given the sentence "She <mask> fun at the <mask> party" and it must predict the words "had" and "birthday" (the other outputs will be ignored). To be more precise, each selected word has an 80% chance of being masked, a 10% chance of being replaced by a

27 Alec Radford et al., "Improving Language Understanding by Generative Pre-Training" (2018).

28 For example, the sentence "Jane had a lot of fun at her friend's birthday party" entails "Jane enjoyed the party", but it is contradicted by "Everyone hated the party" and it is unrelated to "The Earth is flat".

29 Jacob Devlin et al., "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding", *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* 1 (2019).

random word (to reduce the discrepancy between pretraining and fine-tuning, since the model will not see <mask> tokens during fine-tuning), and a 10% chance of being left alone (to bias the model toward the correct answer).

Next sentence prediction (NSP)
: The model is trained to predict whether two sentences are consecutive or not. For example, it should predict that "The dog sleeps" and "It snores loudly" are consecutive sentences, while "The dog sleeps" and "The Earth orbits the Sun" are not consecutive. Later research showed that NSP was not as important as was initially thought, so it was dropped in most later architectures.

The model is trained on these two tasks simultaneously (see Figure 16-11). For the NSP task, the authors inserted a class token (<CLS>) at the start of every input, and the corresponding output token represents the model's prediction: sentence B follows sentence A, or it does not. The two input sentences are concatenated, separated only by a special separation token (<SEP>), and they are fed as input to the model. To help the model know which sentence each input token belongs to, a *segment embedding* is added on top of each token's positional embeddings: there are just two possible segment embeddings, one for sentence A and one for sentence B. For the MLM task, some input words are masked (as we just saw) and the model tries to predict what those words were. The loss is only computed on the NSP prediction and the masked tokens, not on the unmasked ones.

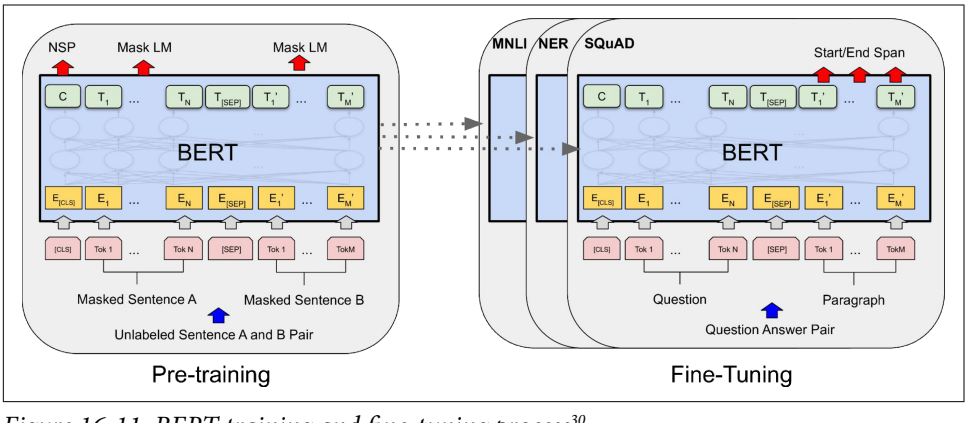

Figure 16-11. BERT training and fine-tuning process[30]

After this unsupervised pretraining phase on a very large corpus of text, the model is then fine-tuned on many different tasks, changing very little for each task. For example, for text classification such as sentiment analysis, all output tokens are

30 This is figure 1 from the paper, reproduced with the kind authorization of the authors.

ignored except for the first one, corresponding to the class token, and a new output layer replaces the previous one, which was just a binary classification layer for NSP.

In February 2019, just a few months after BERT was published, Alec Radford, Jeffrey Wu, and other OpenAI researchers published the GPT-2 paper,[31] which proposed a very similar architecture to GPT, but larger still (with over 1.5 billion parameters!). The researchers showed that the new and improved GPT model could perform *zero-shot learning* (ZSL), meaning it could achieve good performance on many tasks without any fine-tuning. This was just the start of a race toward larger and larger models: Google's Switch Transformers[32] (introduced in January 2021) used 1 trillion parameters, and soon much larger models came out, such as the Wu Dao 2.0 model by the Beijing Academy of Artificial Intelligence (BAII), announced in June 2021.

An unfortunate consequence of this trend toward gigantic models is that only well-funded organizations can afford to train such models: it can easily cost hundreds of thousands of dollars or more. And the energy required to train a single model corresponds to an American household's electricity consumption for several years; it's not eco-friendly at all. Many of these models are just too big to even be used on regular hardware: they wouldn't fit in RAM, and they would be horribly slow. Lastly, some are so costly that they are not released publicly.

Luckily, ingenious researchers are finding new ways to downsize transformers and make them more data-efficient. For example, the DistilBERT model,[33] introduced in October 2019 by Victor Sanh et al. from Hugging Face, is a small and fast transformer model based on BERT. It is available on Hugging Face's excellent model hub, along with thousands of others—you'll see an example later in this chapter.

DistilBERT was trained using *distillation* (hence the name): this means transferring knowledge from a teacher model to a student one, which is usually much smaller than the teacher model. This is typically done by using the teacher's predicted probabilities for each training instance as targets for the student. Surprisingly, distillation often works better than training the student from scratch on the same dataset as the teacher! Indeed, the student benefits from the teacher's more nuanced labels.

Many more transformer architectures came out after BERT, almost on a monthly basis, often improving on the state of the art across all NLP tasks: XLNet (June 2019), RoBERTa (July 2019), StructBERT (August 2019), ALBERT (September 2019), T5 (October 2019), ELECTRA (March 2020), GPT3 (May 2020), DeBERTa (June 2020),

31 Alec Radford et al., "Language Models Are Unsupervised Multitask Learners" (2019).

32 William Fedus et al., "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity" (2021).

33 Victor Sanh et al., "DistilBERT, A Distilled Version of Bert: Smaller, Faster, Cheaper and Lighter", arXiv preprint arXiv:1910.01108 (2019).

Switch Transformers (January 2021), Wu Dao 2.0 (June 2021), Gopher (December 2021), GPT-NeoX-20B (February 2022), Chinchilla (March 2022), OPT (May 2022), and the list goes on and on. Each of these models brought new ideas and techniques,[34] but I particularly like the T5 paper[35] by Google researchers: it frames all NLP tasks as text-to-text, using an encoder–decoder transformer. For example, to translate "I like soccer" to Spanish, you can just call the model with the input sentence "translate English to Spanish: I like soccer" and it outputs "me gusta el fútbol". To summarize a paragraph, you just enter "summarize:" followed by the paragraph, and it outputs the summary. For classification, you only need to change the prefix to "classify:" and the model outputs the class name, as text. This simplifies using the model, and it also makes it possible to pretrain it on even more tasks.

Last but not least, in April 2022, Google researchers used a new large-scale training platform named *Pathways* (which we will briefly discuss in Chapter 19) to train a humongous language model named the *Pathways Language Model* (PaLM),[36] with a whopping 540 billion parameters, using over 6,000 TPUs. Other than its incredible size, this model is a standard transformer, using decoders only (i.e., with masked multi-head attention layers), with just a few tweaks (see the paper for details). This model achieved incredible performance on all sorts of NLP tasks, particularly in natural language understanding (NLU). It's capable of impressive feats, such as explaining jokes, giving detailed step-by-step answers to questions, and even coding. This is in part due to the model's size, but also thanks to a technique called *Chain of thought prompting*,[37] which was introduced a couple months earlier by another team of Google researchers.

In question answering tasks, regular prompting typically includes a few examples of questions and answers, such as: "Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now? A: 11." The prompt then continues with the actual question, such as "Q: John takes care of 10 dogs. Each dog takes .5 hours a day to walk and take care of their business. How many hours a week does he spend taking care of dogs? A:", and the model's job is to append the answer: in this case, "35."

34 Mariya Yao summarized many of these models in this post: *https://homl.info/yaopost*.

35 Colin Raffel et al., "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer", arXiv preprint arXiv:1910.10683 (2019).

36 Aakanksha Chowdhery et al., "PaLM: Scaling Language Modeling with Pathways", arXiv preprint arXiv:2204.02311 (2022).

37 Jason Wei et al., "Chain of Thought Prompting Elicits Reasoning in Large Language Models", arXiv preprint arXiv:2201.11903 (2022).