The Missing Bits

For the sake of concision, this volume skips many important topics, in particular:

Recurrent Neural Networks

Before attention models showed greater performance, Recurrent Neural Networks (RNN) were the standard approach for dealing with temporal sequences such as text or sound samples. These architectures possess an internal hidden state that gets updated each time a component of the sequence is processed. Their main components are layers such as LSTM [Hochreiter and Schmidhuber, 1997] or GRU [Cho et al., 2014].

Training a recurrent architecture amounts to unfolding it in time, which results in a long composition of operators. This has historically prompted the design of key techniques now used for deep architectures such as rectifiers and gating, a form of skip connections which are mod-

ulated dynamically.

One of the key drawbacks of traditional recurrent architectures is that the structure of the computation $x_{t+1} = f(x_t)$ imposes to process the input sequence serially, which takes a time proportional to T. In contrast, transformers, for instance, can take advantage of parallel computation, resulting in a constant time if enough computing units are available.

This is addressed by architectures such as QRNN [Bradbury et al., 2016], S4 [Gu et al., 2021], or Mamba [Gu and Dao, 2023], whose recurrent operations are affine so that the f^t themselves, and consequently the $x_t = f^t(x_0)$, can be computed in parallel, resulting in a constant time if f does not depend on t and $\log T$ otherwise, again if enough parallel computing units are available.

Autoencoder

An autoencoder is a model that maps an input signal, possibly of high dimension, to a low-dimension latent representation, and then maps it back to the original signal, ensuring that information has been preserved. We saw it in § 6.1 for denoising, but it can also be used to automatically discover a meaningful low-dimension

parameterization of the data manifold.

The Variational Autoencoder (VAE) proposed by Kingma and Welling [2013] is a generative model with a similar structure. It imposes, through the loss, a pre-defined distribution on the latent representation. This allows, after training, the generation of new samples by sampling the latent representation according to this imposed distribution and then mapping back through the decoder.

Generative Adversarial Networks

Another approach to density modeling is the Generative Adversarial Networks (GAN) introduced by Goodfellow et al. [2014]. This method combines a generator, which takes a random input following a fixed distribution as input and produces a structured signal such as an image, and a discriminator, which takes a sample as input and predicts whether it comes from the training set or if it was generated by the generator.

Training optimizes the discriminator to minimize a standard cross-entropy loss, and the generator to maximize the discriminator's loss. It can be shown that, at equilibrium, the gener-

ator produces samples indistinguishable from real data. In practice, when the gradient flows through the discriminator to the generator, it informs the latter about the cues that the discriminator uses that need to be addressed.

Graph Neural Networks

Many applications require processing signals which are not organized regularly on a grid. For instance, proteins, 3D meshes, geographic locations, or social interactions are more naturally structured as graphs. Standard convolutional networks or even attention models are poorly adapted to process such data, and the tool of choice for such a task is Graph Neural Networks (GNN) [Scarselli et al., 2009].

These models are composed of layers that compute activations at each vertex by combining linearly the activations located at its immediate neighboring vertices. This operation is very similar to a standard convolution, except that the data structure does not reflect any geometrical information associated with the feature vectors they carry.

Self-supervised training

As stated in § 7.1, even though they are trained only to predict the next word, Large Language Models trained on large unlabeled datasets such as GPT (see § 5.3) are able to solve various tasks, such as identifying the grammatical role of a word, answering questions, or even translating from one language to another [Radford et al., 2019].

Such models constitute one category of a larger class of methods that fall under the name of self-supervised learning, and try to take advantage of unlabeled datasets [Balestriero et al., 2023].

The key principle of these methods is to define a task that does not require labels but necessitates feature representations which are useful for the real task of interest, for which a small labeled dataset exists. In computer vision, for instance, image features can be optimized so that they are invariant to data transformations that do not change the semantic content of the image, while being statistically uncorrelated [Zbontar et al., 2021].

In both NLP and computer vision, a powerful generic strategy is to train a model to recover parts of the signal that have been masked [Devlin

et al., 2018; Zhou et al., 2021].