A **recurrent neural network** (**RNN**) is one of the two broad types of artificial neural network, characterized by direction of the flow of information between its layers. In contrast to the uni-directional feedforward neural network, it is a bi-directional artificial neural network, meaning that it allows the output from some nodes to affect subsequent input to the same nodes. Their ability to use internal state (memory) to process arbitrary sequences of inputs makes them applicable to tasks such as unsegmented, connected handwriting recognition[4] or speech recognition. The term "recurrent neural network" is used to refer to the class of networks with an infinite impulse response, whereas "convolutional neural network" refers to the class of finite impulse response. Both classes of networks exhibit temporal dynamic behavior. A finite impulse recurrent network is a directed acyclic graph that can be unrolled and replaced with a strictly feedforward neural network, while an infinite impulse recurrent network is a directed cyclic graph that cannot be unrolled.

Additional stored states and the storage under direct control by the network can be added to both infinite-impulse and finite-impulse networks. Another network or graph can also replace the storage if that incorporates time delays or has feedback loops. Such controlled states are referred to as gated states or gated memory and are part of long short-term memory networks (LSTMs) and gated recurrent units. This is also called Feedback Neural Network (FNN). Recurrent neural networks are theoretically Turing complete and can run arbitrary programs to process arbitrary sequences of inputs.

Architectures

RNNs come in many variants.

Fully recurrent

Fully recurrent neural networks (FRNN) connect the outputs of all neurons to the inputs of all neurons. This is the most general neural network topology because all other topologies can be represented by setting some connection weights to zero to simulate the lack of connections between those neurons. The illustration to the right may be misleading to many because practical neural network topologies are frequently organized in "layers" and the drawing gives that appearance. However, what appears to be layers are, in

fact, different steps in time of the same fully recurrent neural network. The left-most item in the illustration shows the recurrent connections as the arc labeled 'v'. It is "unfolded" in time to produce the appearance of layers.

Elman networks and Jordan networks

An Elman network is a three-layer network (arranged horizontally as *x*, *y*, and *z* in the illustration) with the addition of a set of context units (*u* in the illustration). The middle (hidden) layer is connected to these context units fixed with a weight of one. At each time step, the input is fed forward and a learning rule is applied. The fixed back-connections save a copy of the previous values of the hidden units in the context units (since they propagate over the connections before the learning rule is applied). Thus the network can maintain a sort of state, allowing it to perform tasks such as sequence-prediction that are beyond the power of a standard multilayer perceptron.

Jordan networks are similar to Elman networks. The context units are fed from the output layer instead of the hidden layer. The context units in a Jordan network are also called the state layer. They have a recurrent connection to themselves.

Elman and Jordan networks are also known as "Simple recurrent networks" (SRN).

Hopfield

The Hopfield network is an RNN in which all connections across layers are equally sized. It requires stationary inputs and is thus not a general RNN, as it does not process sequences of patterns. However, it guarantees that it will converge. If the connections are trained using Hebbian learning, then the Hopfield network can perform as robust content-addressable memory, resistant to connection alteration.

Bidirectional associative memory

Introduced by Bart Kosko, a bidirectional associative memory (BAM) network is a variant of a Hopfield network that stores associative data as a vector.

The bi-directionality comes from passing information through a matrix and its transpose. Typically, bipolar encoding is preferred to binary encoding of the associative pairs. Recently, stochastic BAM models using Markov stepping were optimized for increased network stability and relevance to real-world applications.

A BAM network has two layers, either of which can be driven as an input to recall an association and produce an output on the other layer.

Echo state

Echo state networks (ESN) have a sparsely connected random hidden layer. The weights of output neurons are the only part of the network that can change (be trained). ESNs are good at reproducing certain time series. A variant for spiking neurons is known as a liquid state machine.

Independently RNN (IndRNN)

The independently recurrent neural network (IndRNN) addresses the gradient vanishing and exploding problems in the traditional fully connected RNN. Each neuron in one layer only receives its own past state as context information (instead of full connectivity to all other neurons in this layer) and thus neurons are independent of each other's history. The gradient backpropagation can be regulated to avoid gradient vanishing and exploding in order to keep long or short-term memory. The cross-neuron information is explored in the next layers. IndRNN can be robustly trained with non-saturated nonlinear functions such as ReLU. Deep networks can be trained using skip connections.

Recursive

A recursive neural network[38] is created by applying the same set of weights recursively over a differentiable graph-like structure by traversing the structure in topological order. Such networks are typically also trained by the reverse mode of automatic differentiation. They can process distributed representations of structure, such as logical terms. A special case of recursive neural networks is the RNN whose structure corresponds to a linear chain. Recursive neural networks have been applied to natural language

processing. The Recursive Neural Tensor Network uses a tensor-based composition function for all nodes in the tree.

Neural history compressor

The neural history compressor is an unsupervised stack of RNNs. At the input level, it learns to predict its next input from the previous inputs. Only unpredictable inputs of some RNN in the hierarchy become inputs to the next higher level RNN, which therefore recomputes its internal state only rarely. Each higher level RNN thus studies a compressed representation of the information in the RNN below. This is done such that the input sequence can be precisely reconstructed from the representation at the highest level.

The system effectively minimizes the description length or the negative logarithm of the probability of the data. Given a lot of learnable predictability in the incoming data sequence, the highest level RNN can use supervised learning to easily classify even deep sequences with long intervals between important events.

It is possible to distill the RNN hierarchy into two RNNs: the "conscious" chunker (higher level) and the "subconscious" automatizer (lower level). Once the chunker has learned to predict and compress inputs that are unpredictable by the automatizer, then the automatizer can be forced in the next learning phase to predict or imitate through additional units the hidden units of the more slowly changing chunker. This makes it easy for the automatizer to learn appropriate, rarely changing memories across long intervals. In turn, this helps the automatizer to make many of its once unpredictable inputs predictable, such that the chunker can focus on the remaining unpredictable events.

A generative model partially overcame the vanishing gradient problem of automatic differentiation or backpropagation in neural networks in 1992. In 1993, such a system solved a "Very Deep Learning" task that required more than 1000 subsequent layers in an RNN unfolded in time.

Second order RNNs

Second-order RNNs use higher order weights w(ijk) instead of the standard w(ij) weights, and states can be a product. This allows a direct mapping to a finite-state machine both in training, stability, and

representation. Long short-term memory is an example of this but has no such formal mappings or proof of stability.

Long short-term memory

Long short-term memory (LSTM) is a deep learning system that avoids the vanishing gradient problem. LSTM is normally augmented by recurrent gates called "forget gates". LSTM prevents backpropagated errors from vanishing or exploding. Instead, errors can flow backward through unlimited numbers of virtual layers unfolded in space. That is, LSTM can learn tasks that require memories of events that happened thousands or even millions of discrete time steps earlier. Problem-specific LSTM-like topologies can be evolved. LSTM works even given long delays between significant events and can handle signals that mix low and high-frequency components.

Many applications use stacks of LSTM RNNs and train them by connectionist temporal classification (CTC) to find an RNN weight matrix that maximizes the probability of the label sequences in a training set, given the corresponding input sequences. CTC achieves both alignment and recognition.

LSTM can learn to recognize context-sensitive languages unlike previous models based on hidden Markov models (HMM) and similar concepts.

Gated recurrent unit

Gated recurrent units (GRUs) are a gating mechanism in recurrent neural networks introduced in 2014. They are used in the full form and several simplified variants. Their performance on polyphonic music modeling and speech signal modeling was found to be similar to that of long short-term memory. They have fewer parameters than LSTM, as they lack an output gate.

Bi-directional

Bi-directional RNNs use a finite sequence to predict or label each element of the sequence based on the element's past and future contexts. This is done by concatenating the outputs of two RNNs, one processing the sequence from left to right, and the other one from right to left. The combined outputs are

the predictions of the teacher-given target signals. This technique has been proven to be especially useful when combined with LSTM RNNs.

Hierarchical recurrent neural network

Hierarchical recurrent neural networks (HRNN) connect their neurons in various ways to decompose hierarchical behavior into useful subprograms. Such hierarchical structures of cognition are present in theories of memory presented by philosopher Henri Bergson, whose philosophical views have inspired hierarchical models.

Hierarchical recurrent neural networks are useful in forecasting, helping to predict disaggregated inflation components of the consumer price index (CPI). The HRNN model leverages information from higher levels in the CPI hierarchy to enhance lower-level predictions. Evaluation of a substantial dataset from the US CPI-U index demonstrates the superior performance of the HRNN model compared to various established inflation prediction methods.

Recurrent multilayer perceptron network

Generally, a recurrent multilayer perceptron network (RMLP network) consists of cascaded subnetworks, each containing multiple layers of nodes. Each subnetwork is feed-forward except for the last layer, which can have feedback connections. Each of these subnets is connected only by feed-forward connections.

Multiple timescales model

A multiple timescales recurrent neural network (MTRNN) is a neural-based computational model that can simulate the functional hierarchy of the brain through self-organization depending on the spatial connection between neurons and on distinct types of neuron activities, each with distinct time properties. With such varied neuronal activities, continuous sequences of any set of behaviors are segmented into reusable primitives, which in turn are flexibly integrated into diverse sequential behaviors. The biological approval of such a type of hierarchy was discussed in the memory-prediction theory of brain function by Hawkins in his book *On Intelligence*. Such a hierarchy also

agrees with theories of memory posited by philosopher Henri Bergson, which have been incorporated into an MTRNN model.

Neural Turing machines

Neural Turing machines (NTMs) are a method of extending recurrent neural networks by coupling them to external memory resources which they can interact with by attentional processes. The combined system is analogous to a Turing machine or Von Neumann architecture but is differentiable end-to-end, allowing it to be efficiently trained with gradient descent.

Differentiable neural computer

Differentiable neural computers (DNCs) are an extension of Neural Turing machines, allowing for the usage of fuzzy amounts of each memory address and a record of chronology.

Neural network pushdown automata

Neural network pushdown automata (NNPDA) are similar to NTMs, but tapes are replaced by analog stacks that are differentiable and trained. In this way, they are similar in complexity to recognizers of context free grammars (CFGs).

Memristive networks

Greg Snider of HP Labs describes a system of cortical computing with memristive nanodevices. The memristors (memory resistors) are implemented by thin film materials in which the resistance is electrically tuned via the transport of ions or oxygen vacancies within the film. DARPA's SyNAPSE project has funded IBM Research and HP Labs, in collaboration with the Boston University Department of Cognitive and Neural Systems (CNS), to develop neuromorphic architectures that may be based on memristive systems. Memristive networks are a particular type of physical neural network that have very similar properties to (Little-)Hopfield networks, as they have continuous dynamics, a limited memory capacity and natural relaxation via the minimization of a function which is asymptotic to the Ising model. In this sense, the dynamics of a memristive circuit have the

advantage compared to a Resistor-Capacitor network to have a more interesting non-linear behavior. From this point of view, engineering analog memristive networks account for a peculiar type of neuromorphic engineering in which the device behavior depends on the circuit wiring or topology. The evolution of these networks can be studied analytically using variations of the Caravelli–Traversa–Di Ventra equation.