

WILDML

AI, DEEP LEARNING, NLP

MENU

DEEP LEARNING GLOSSARY

This glossary is work in progress and I am planning to continuously update it. If you find a mistake or think an important term is missing, please let me know in the comments or [via email](#).

Deep Learning terminology can be quite overwhelming to newcomers. This glossary tries to define commonly used terms and link to original references and additional resources to help readers dive deeper into a specific topic.

The boundary between what is Deep Learning vs. “general” Machine Learning terminology is quite fuzzy. I am trying to keep the glossary specific to Deep Learning, but these decisions are somewhat arbitrary. For example, I am not including “cross-validation” here because it’s a generic technique used all across Machine Learning. However, I’ve decided to include terms such as [softmax](#) or [word2vec](#) because they are often associated with Deep Learning even though they are not Deep Learning techniques.

ACTIVATION FUNCTION

To allow Neural Networks to learn complex decision boundaries, we apply a nonlinear activation function to some of its layers. Commonly used functions include [sigmoid](#), [tanh](#), [ReLU \(Rectified Linear Unit\)](#) and variants of these.

ADADELTA

Adadelata is a gradient descent based learning algorithm that adapts the learning rate per parameter over time. It was proposed as an improvement over **Adagrad**, which is more sensitive to hyperparameters and may decrease the learning rate too aggressively. Adadelata It is similar to **rmsprop** and can be used instead of vanilla **SGD**.

- **ADADELTA: An Adaptive Learning Rate Method**
- **Stanford CS231n: Optimization Algorithms**
- **An overview of gradient descent optimization algorithms**

ADAGRAD

Adagrad is an adaptive learning rate algorithms that keeps track of the squared gradients over time and automatically adapts the learning rate per-parameter. It can be used instead of vanilla SGD and is particularly helpful for sparse data, where it assigns a higher learning rate to infrequently updated parameters.

- **Adaptive Subgradient Methods for Online Learning and Stochastic Optimization**
- **Stanford CS231n: Optimization Algorithms**
- **An overview of gradient descent optimization algorithms**

ADAM

Adam is an adaptive learning rate algorithm similar to **rmsprop**, but updates are directly estimated using a running average of the first and second moment of the gradient and also include a bias correction term.

- **Adam: A Method for Stochastic Optimization**
- **An overview of gradient descent optimization algorithms**

AFFINE LAYER

A fully-connected layer in a Neural Network. Affine means that each neuron in the previous layer is connected to each neuron in the current layer. In many ways, this is the “standard” layer of a Neural Network. Affine layers are often added on top of the outputs of **Convolutional Neural Networks** or **Recurrent Neural Networks** before making a final

prediction. An affine layer is typically of the form $y = f(Wx + b)$ where x are the layer inputs, W the parameters, b a bias vector, and f a nonlinear **activation function**

ATTENTION MECHANISM

Attention Mechanisms are inspired by human visual attention, the ability to focus on specific parts of an image. Attention mechanisms can be incorporated in both Language Processing and Image Recognition architectures to help the network learn what to “focus” on when making predictions.

- **Attention and Memory in Deep Learning and NLP**

ALEXNET

Alexnet is the name of the Convolutional Neural Network architecture that won the ILSVRC 2012 competition by a large margin and was responsible for a resurgence of interest in CNNs for Image Recognition. It consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. Alexnet was introduced in **ImageNet Classification with Deep Convolutional Neural Networks**.

AUTOENCODER

An Autoencoder is a Neural Network model whose goal is to predict the input itself, typically through a “bottleneck” somewhere in the network. By introducing a bottleneck, we force the network to learn a lower-dimensional representation of the input, effectively compressing the input into a good representation. Autoencoders are related to PCA and other dimensionality reduction techniques, but can learn more complex mappings due to their nonlinear nature. A wide range of autoencoder architectures exist, including **Denoising Autoencoders**, **Variational Autoencoders**, or **Sequence Autoencoders**.

AVERAGE-POOLING

Average-Pooling is a **pooling** technique used in Convolutional Neural Networks for Image Recognition. It works by sliding a window over patches of features, such as pixels,

and taking the average of all values within the window. It compresses the input representation into a lower-dimensional representation.

BACKPROPAGATION

Backpropagation is an algorithm to efficiently calculate the gradients in a Neural Network, or more generally, a feedforward computational graph. It boils down to applying the chain rule of differentiation starting from the network output and propagating the gradients backward. The first uses of backpropagation go back to Vapnik in the 1960's, but **Learning representations by back-propagating errors** is often cited as the source.

- **Calculus on Computational Graphs: Backpropagation**

BACKPROPAGATION THROUGH TIME (BPTT)

Backpropagation Through Time (**paper**) is the **Backpropagation algorithm** applied to Recurrent Neural Networks (RNNs). BPTT can be seen as the standard backpropagation algorithm applied to an RNN, where each time step represents a layer and the parameters are shared across layers. Because an RNN shares the same parameters across all time steps, the errors at one time step must be backpropagated “through time” to all previous time steps, hence the name. When dealing with long sequences (hundreds of inputs), a truncated version of BPTT is often used to reduce the computational cost. Truncated BPTT stops backpropagating the errors after a fixed number of steps.

- **Backpropagation Through Time: What It Does and How to Do It**

BATCH NORMALIZATION

Batch Normalization is a technique that normalizes layer inputs per mini-batch. It speeds up training, allows for the usage of higher learner rates, and can act as a regularizer. Batch Normalization has been found to be very effective for Convolutional and Feedforward Neural Networks but hasn't been successfully applied to Recurrent Neural Networks.

- **Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift**
- **Batch Normalized Recurrent Neural Networks**

BIDIRECTIONAL RNN

A Bidirectional Recurrent Neural Network is a type of Neural Network that contains two **RNNs** going into different directions. The forward RNN reads the input sequence from start to end, while the backward RNN reads it from end to start. The two RNNs are stacked on top of each others and their states are typically combined by appending the two vectors. Bidirectional RNNs are often used in Natural Language problems, where we want to take the context from both before and after a word into account before making a prediction.

- **Bidirectional Recurrent Neural Networks**

CAFFE

Caffe is a deep learning framework developed by the Berkeley Vision and Learning Center. Caffe is particularly popular and performant for vision tasks and **CNN** models.

CATEGORICAL CROSS-ENTROPY LOSS

The categorical cross-entropy loss is also known as the negative log likelihood. It is a popular loss function for categorization problems and measures the similarity between two probability distributions, typically the true labels and the predicted labels. It is given by $L = -\sum (y * \log(y_prediction))$ where y is the probability distribution of true labels (typically a one-hot vector) and $y_prediction$ is the probability distribution of the predicted labels, often coming from a **softmax**.

CHANNEL

Input data to Deep Learning models can have multiple channels. The canonical examples are images, which have red, green and blue color channels. A image can be represented as a 3-dimensional Tensor with the dimensions corresponding to channel,

height, and width. Natural Language data can also have multiple channels, in the form of different types of **embeddings** for example.

CONVOLUTIONAL NEURAL NETWORK (CNN, CONVNET)

A CNN uses **convolutions** to connected extract features from local regions of an input. Most CNNs contain a combination of convolutional, **pooling** and **affine** layers. CNNs have gained popularity particularly through their excellent performance on visual recognition tasks, where they have been setting the state of the art for several years.

- **Stanford CS231n class – Convolutional Neural Networks for Visual Recognition**
- **Understanding Convolutional Neural Networks for NLP**

DEEP BELIEF NETWORK (DBN)

DBNs are a type of probabilistic graphical model that learn a hierarchical representation of the data in an unsupervised manner. DBNs consist of multiple hidden layers with connections between neurons in each successive pair of layers. DBNs are built by stacking multiple **RBNs** on top of each other and training them one by one.

- **A fast learning algorithm for deep belief nets**

DEEP DREAM

A technique invented by Google that tries to distill the knowledge captured by a deep Convolutional Neural Network. The technique can generate new images, or transform existing images and give them a dreamlike flavor, especially when applied recursively.

- **Deep Dream on Github**
- **Inceptionism: Going Deeper into Neural Networks**

DROPOUT

Dropout is a regularization technique for Neural Networks that prevents overfitting. It prevents neurons from co-adapting by randomly setting a fraction of them to 0 at each training iteration. Dropout can be interpreted in various ways, such as randomly sampling from an exponential number of different networks. Dropout layers first gained

popularity through their use in **CNNs**, but have since been applied to other layers, including input embeddings or recurrent networks.

- **Dropout: A Simple Way to Prevent Neural Networks from Overfitting**
- **Recurrent Neural Network Regularization**

EMBEDDING

An embedding maps an input representation, such as a word or sentence, into a vector. A popular type of embedding are word embeddings such as **word2vec** or **GloVe**. We can also embed sentences, paragraphs or images. For example, by mapping images and their textual descriptions into a common embedding space and minimizing the distance between them, we can match labels with images. Embeddings can be learned explicitly, such as in **word2vec**, or as part of a supervised task, such as Sentiment Analysis. Often, the input layer of a network is initialized with pre-trained embeddings, which are then **fine-tuned** to the task at hand.

EXPLODING GRADIENT PROBLEM

The Exploding Gradient Problem is the opposite of the **Vanishing Gradient Problem**. In Deep Neural Networks gradients may explode during backpropagation, resulting number overflows. A common technique to deal with exploding gradients is to perform **Gradient Clipping**.

- **On the difficulty of training recurrent neural networks**

FINE-TUNING

Fine-Tuning refers to the technique of initializing a network with parameters from another task (such as an unsupervised training task), and then updating these parameters based on the task at hand. For example, NLP architecture often use pre-trained word embeddings like **word2vec**, and these word embeddings are then updated during training based for a specific task like Sentiment Analysis.

GRADIENT CLIPPING

Gradient Clipping is a technique to prevent **exploding gradients** in very deep networks, typically Recurrent Neural Networks. There exist various ways to perform gradient clipping, but the a common one is to normalize the gradients of a parameter vector when its L2 norm exceeds a certain threshold according to $\text{new_gradients} = \text{gradients} * \text{threshold} / \text{l2_norm}(\text{gradients})$.

- **On the difficulty of training recurrent neural networks**

GLOVE

GloVe is an unsupervised learning algorithm for obtaining vector representations (**embeddings**) for words. GloVe vectors serve the same purpose as word2vec but have different vector representations due to being trained on co-occurrence statistics.

- **GloVe: Global Vectors for Word Representation**

GOOGLELENET

The name of the Convolutional Neural Network architecture that won the ILSVRC 2014 challenge. The network uses **Inception modules** to reduce the parameters and improve the utilization of the computing resources inside the network.

- **Going Deeper with Convolutions**

GRU

The Gated Recurrent Unit is a simplified version of an LSTM unit with fewer parameters. Just like an LSTM cell, it uses a gating mechanism to allow RNNs to efficiently learn long-range dependency by preventing the **vanishing gradient problem**. The GRU consists of a reset and update gate that determine which part of the old memory to keep vs. update with new values at the current time step.

- **Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation**
- **Recurrent Neural Network Tutorial, Part 4 – Implementing a GRU/LSTM RNN with Python and Theano**

HIGHWAY LAYER

A Highway Layer ([paper](#)) is a type of Neural Network layer that uses a gating mechanism to control the information flow through a layer. Stacking multiple Highway Layers allows for training of very deep networks. Highway Layers work by learning a gating function that chooses which parts of the inputs to pass through and which parts to pass through a transformation function, such as a standard [affine](#) layer for example. The basic formulation of a Highway Layer is $T * h(x) + (1 - T) * x$, where T is the learned gating function with values between 0 and 1, $h(x)$ is an arbitrary input transformation and x is the input. Note that all of these must have the same size.

ICML

The [International Conference for Machine Learning](#), a top-tier machine learning conference.

ILSVRC

The [ImageNet Large Scale Visual Recognition Challenge](#) evaluates algorithms for object detection and image classification at large scale. It is the most popular academic challenge in computer vision. Over the past years, Deep Learning techniques have led to a significant reduction in error rates, from 30% to less than 5%, beating human performance on several classification tasks.

INCEPTION MODULE

Inception Modules are used in Convolutional Neural Networks to allow for more efficient computation and deeper Networks through a dimensionality reduction with stacked 1×1 convolutions.

- [Going Deeper with Convolutions](#)

KERAS

[Keras](#) is a Python-based Deep Learning library that includes many high-level building blocks for deep Neural Networks. It can run on top of either [TensorFlow](#) or [Theano](#).

LSTM

Long Short-Term Memory networks were invented to prevent the **vanishing gradient problem** in Recurrent Neural Networks by using a memory gating mechanism. Using LSTM units to calculate the hidden state in an RNN we help to the network to efficiently propagate gradients and learn long-range dependencies.

- **Long Short-Term Memory**
- **Understanding LSTM Networks**
- **Recurrent Neural Network Tutorial, Part 4 – Implementing a GRU/LSTM RNN with Python and Theano**

MAX-POOLING

A **pooling** operations typically used in Convolutional Neural Networks. A max-pooling layer selects the maximum value from a patch of features. Just like a convolutional layer, pooling layers are parameterized by a window (patch) size and stride size. For example, we may slide a window of size 2×2 over a 10×10 feature matrix using stride size 2, selecting the max across all 4 values within each window, resulting in a new 5×5 feature matrix. Pooling layers help to reduce the dimensionality of a representation by keeping only the most salient information, and in the case of image inputs, they provide basic invariance to translation (the same maximum values will be selected even if the image is shifted by a few pixels). Pooling layers are typically inserted between successive convolutional layers.

MNIST

The **MNIST data set** is the perhaps most commonly used Image Recognition dataset. It consists of 60,000 training and 10,000 test examples of handwritten digits. Each image is 28×28 pixels large. State of the art models typically achieve accuracies of 99.5% or higher on the test set.

MOMENTUM

Momentum is an extension to the Gradient Descent Algorithm that accelerates or damps the parameter updates. In practice, including a momentum term in the gradient descent updates leads to better convergence rates in Deep Networks.

- **Learning representations by back-propagating errors**

MULTILAYER PERCEPTRON (MLP)

A Multilayer Perceptron is a Feedforward Neural Network with multiple fully-connected layers that use nonlinear **activation functions** to deal with data which is not linearly separable. An MLP is the most basic form of a multilayer Neural Network, or a deep Neural Networks if it has more than 2 layers.

NEGATIVE LOG LIKELIHOOD (NLL)

See **Categorical Cross Entropy Loss**.

NEURAL MACHINE TRANSLATION (NMT)

An NMT system uses Neural Networks to translate between languages, such as English and French. NMT systems can be trained end-to-end using bilingual corpora, which differs from traditional Machine Translation systems that require hand-crafted features and engineering. NMT systems are typically implemented using encoder and decoder recurrent neural networks that encode a source sentence and produce a target sentence, respectively.

- **Sequence to sequence learning with neural networks**
- **Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation**

NEURAL TURING MACHINE (NTM)

NTMs are Neural Network architectures that can infer simple algorithms from examples. For example, a NTM may learn a sorting algorithm through example inputs and outputs. NTMs typically learn some form of memory and attention mechanism to deal with state during program execution.

- **Neural Turing Machines**

NONLINEARITY

See [Activation Function](#).

NOISE-CONTRASTIVE ESTIMATION (NCE)

Noise-contrastive estimation is a sampling loss typically used to train classifiers with a large output vocabulary. Calculating the [softmax](#) over a large number of possible classes is prohibitively expensive. Using NCE, we can reduce the problem to binary classification problem by training the classifier to discriminate between samples from the “real” distribution and an artificially generated noise distribution.

- [Noise-contrastive estimation: A new estimation principle for unnormalized statistical models](#)
- [Learning word embeddings efficiently with noise-contrastive estimation](#)

POOLING

See [Max-Pooling](#) or [Average-Pooling](#).

RESTRICTED BOLTZMANN MACHINE (RBN)

RBMs are a type of probabilistic graphical model that can be interpreted as a stochastic artificial neural network. RBNs learn a representation of the data in an unsupervised manner. An RBN consists of visible and hidden layer, and connections between binary neurons in each of these layers. RBNs can be efficiently trained using [Contrastive Divergence](#), an approximation of gradient descent.

- [Chapter 6: Information Processing in Dynamical Systems: Foundations of Harmony Theory](#)
- [An Introduction to Restricted Boltzmann Machines](#)

RECURRENT NEURAL NETWORK (RNN)

A RNN models sequential interactions through a hidden state, or memory. It can take up to N inputs and produce up to N outputs. For example, an input sequence may be a sentence with the outputs being the part-of-speech tag for each word (N-to-N). An input could be a sentence, and the output a sentiment classification of the sentence (N-to-1). An input could be a single image, and the output could be a sequence of words

corresponding to the description of an image (1-to-N). At each time step, an RNN calculates a new hidden state (“memory”) based on the current input and the previous hidden state. The “recurrent” stems from the facts that at each step the same parameters are used and the network performs the same calculations based on different inputs.

- [Understanding LSTM Networks](#)
- [Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs](#)

RECURSIVE NEURAL NETWORK

Recursive Neural Networks are a generalization of [Recurrent Neural Networks](#) to a tree-like structure. The same weights are applied at each recursion. Just like RNNs, Recursive Neural Networks can be trained end-to-end using backpropagation. While it is possible to learn the tree structure as part of the optimization problem, Recursive Neural Networks are often applied to problem that already have a predefined structure, like a parse tree in Natural Language Processing.

- [Parsing Natural Scenes and Natural Language with Recursive Neural Networks](#)

RELU

Short for Rectified Linear Unit(s). ReLUs are often used as [activation functions](#) in Deep Neural Networks. They are defined by $f(x) = \max(0, x)$. The advantages of ReLUs over functions like \tanh include that they tend to be sparse (their activation easily be set to 0), and that they suffer less from the [vanishing gradient problem](#). ReLUs are the most commonly used activation function in Convolutional Neural Networks. There exist several variations of ReLUs, such as [Leaky ReLUs](#), [Parametric ReLU \(PReLU\)](#) or a smoother [softplus](#) approximation.

- [Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification](#)
- [Rectifier Nonlinearities Improve Neural Network Acoustic Models](#)
- [Rectified Linear Units Improve Restricted Boltzmann Machines](#)

RESNET

Deep Residual Networks won the ILSVRC 2015 challenge. These networks work by introducing shortcut connection across stacks of layers, allowing the optimizer to learn “easier” residual mappings instead of the more complicated original mappings. These shortcut connections are similar to **Highway Layers**, but they are data-independent and don’t introduce additional parameters or training complexity. ResNets achieved a 3.57% error rate on the ImageNet test set.

- **Deep Residual Learning for Image Recognition**

RMSPROP

RMSProp is a gradient-based optimization algorithm. It is similar to **Adagrad**, but introduces an additional decay term to counteract Adagrad’s rapid decrease in learning rate.

- **Neural Networks for Machine Learning Lecture 6a**
- **Stanford CS231n: Optimization Algorithms**
- **An overview of gradient descent optimization algorithms**

SEQ2SEQ

A Sequence-to-Sequence model reads a sequence (such as a sentence) as an input and produces another sequence as an output. It differs from a standard **RNN** in that the input sequence is completely read before the network starts producing any output. Typically, seq2seq models are implemented using two RNNs, functioning as encoders and decoders. **Neural Machine Translation** is a typical example of a seq2seq model.

- **Sequence to Sequence Learning with Neural Networks**

SGD

Stochastic Gradient Descent (**Wikipedia**) is a gradient-based optimization algorithm that is used to learn network parameters during the training phase. The gradients are typically calculated using the **backpropagation** algorithm. In practice, people use the minibatch version of SGD, where the parameter updates are performed based on a

batch instead of a single example, increasing computational efficiency. Many extensions to vanilla SGD exist, including **Momentum**, **Adagrad**, **rmsprop**, **Adadelata** or **Adam**.

- **Adaptive Subgradient Methods for Online Learning and Stochastic Optimization**
- **Stanford CS231n: Optimization Algorithms**
- **An overview of gradient descent optimization algorithms**

SOFTMAX

The **softmax function** is typically used to convert a vector of raw scores into class probabilities at the output layer of a Neural Network used for classification. It normalizes the scores by exponentiating and dividing by a normalization constant. If we are dealing with a large number of classes, a large vocabulary in Machine Translation for example, the normalization constant is expensive to compute. There exist various alternatives to make the computation more efficient, including **Hierarchical Softmax** or using a sampling-based loss such as **NCE**.

TENSORFLOW

TensorFlow is an open source C++/Python software library for numerical computation using data flow graphs, particularly Deep Neural Networks. It was created by Google. In terms of design, it is most similar to **Theano**, and lower-level than **Caffe** or **Keras**.

THEANO

Theano is a Python library that allows you to define, optimize, and evaluate mathematical expressions. It contains many building blocks for deep neural networks. Theano is a low-level library similar to **Tensorflow**. Higher-level libraries include **Keras** and **Caffe**.

VANISHING GRADIENT PROBLEM

The vanishing gradient problem arises in very deep Neural Networks, typically Recurrent Neural Networks, that use activation functions whose gradients tend to be small (in the range of 0 from 1). Because these small gradients are multiplied during backpropagation, they tend to “vanish” throughout the layers, preventing the network from learning long-range dependencies. Common ways to counter this problem is to

use activation functions like **ReLU**s that do not suffer from small gradients, or use architectures like **LSTMs** that explicitly combat vanishing gradients. The opposite of this problem is called the **exploding gradient problem**.

- **On the difficulty of training recurrent neural networks**

VGG

VGG refers to convolutional neural network model that secured the first and second place in the 2014 ImageNet localization and classification tracks, respectively. The VGG model consist of 16–19 weight layers and uses small convolutional filters of size 3×3 and 1×1.

- **Very Deep Convolutional Networks for Large-Scale Image Recognition**

WORD2VEC

word2vec is an algorithm and **tool** to learn word **embeddings** by trying to predict the context of words in a document. The resulting word vectors have some interesting properties, for example `vector('queen') ≈ vector('king') - vector('man') + vector('woman')`. Two different objectives can be used to learn these embeddings: The Skip-Gram objective tries to predict a context from on a word, and the CBOW objective tries to predict a word from its context.

- **Efficient Estimation of Word Representations in Vector Space**
- **Distributed Representations of Words and Phrases and their Compositionality**
- **word2vec Parameter Learning Explained**

- CONNECT -

- RECENT POSTS -

[Deep Learning for Chatbots, Part 1 – Introduction](#)

[Attention and Memory in Deep Learning and NLP](#)

[Implementing a CNN for Text Classification in TensorFlow](#)

[Understanding Convolutional Neural Networks for NLP](#)

[Recurrent Neural Network Tutorial, Part 4 – Implementing a GRU/LSTM RNN with Python and Theano](#)

[Recurrent Neural Networks Tutorial, Part 3 – Backpropagation Through Time and Vanishing Gradients](#)

[Recurrent Neural Networks Tutorial, Part 2 – Implementing a RNN with Python, Numpy and Theano](#)

[Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs](#)

- LINKS -

[Home](#)

[About](#)

[NLP Consulting](#)

- ARCHIVES -

[April 2016](#)

[January 2016](#)

[December 2015](#)

[November 2015](#)

[October 2015](#)

September 2015

- CATEGORIES -

Conversational Agents

Convolutional Neural Networks

Deep Learning

GPU

Language Modeling

Memory

Neural Networks

NLP

Recurrent Neural Networks

RNNs

- SUBSCRIBE TO BLOG VIA EMAIL -

Enter your email address to subscribe to this blog and receive notifications of new posts by email.

SUBSCRIBE

- META -

Log in

Entries RSS

[Comments RSS](#)

[WordPress.org](#)

PROUDLY POWERED BY WORDPRESS | THEME: SELA BY WORDPRESS.COM.