DOWNLOAD PDF NEURAL NETWORK MODELS

Chapter 1 : A new convolutional neural network model to detect abuse and incivility on Twitter

A neural network is a simplified model of the way the human brain processes information. It works by simulating a large number of interconnected processing units that resemble abstract versions of neurons.

The draft can be found here: Paper A topic that I have been interested in for a long time is forecasting mortality rates, perhaps because this is one of the interesting intersections of statistics and these days, machine learning, and the field of actuarial science in life insurance. Several methods to model mortality rates over time have been proposed, ranging from the relatively simple method of extrapolating mortality rates directly using time series, to more complicated statistical approaches. One of the most famous of these is the Lee-Carter method, which models mortality as an average mortality rate that changes over time. The change over time is governed by a time-based mortality index, which is common to all ages, and an age-specific rate of change factor: How are these quantities derived? There are two methods prominent in the literature â€" applying Principal Components Analysis, or Generalized Non-linear Models, which are different from GLMs in the sense that the user can specify non-additive relationship between two or more terms. In the current age of big data, relatively high quality mortality data spanning an extended period are available for many countries from the excellent Human Mortality Database, which is a resource that anyone with an interest in the study of mortality can benefit from. The challenge, though, is how to model all of these data simultaneously to improve mortality forecasts? While some extensions of basic models like the Lee-Carter model have been proposed, these rely on assumptions that might not necessarily be applicable in the case of large scale mortality forecasting. For example, some of the common multi-population mortality models rely on the assumption of a common mortality trend for all countries, which is likely not the case. In the paper, we tackle this problem in a novel way â€" feed all the variables to a deep neural network and let it figure out how exactly to model the mortality rates over time. This speaks to the idea of representation learning that is central to modern deep learning, which is that many datasets, such as large collections of images as in the ImageNet dataset, are too complicated to model by hand-engineering features, or it is too time consuming to perform the modelling. Rather, the strategy in deep learning is to define a neural network architecture that expresses useful priors about the data, and allow the network to learn how the raw data relates to the problem at hand. In the example of modelling mortality rates, we use two architectural elements that are common in applications of neural networks to tabular data: We use a deep network, in other words the network consists of multiple layers of variables that expresses the prior that complex features can be represented by a hierarchy of simpler representations learned in the model. Instead of using one-hot encoding to signify to the network when we are modelling a particular country, or gender, we use embedding layers. When applied to many categories, one-hot encoding produces a high-dimensional feature vector that is sparse i. Even if there is enough data, as in our case of mortality rates, each parameter is learned in isolation, and the estimated parameters do not share information, unless the modeller explicitly chooses to use something like credibility or mixed models. The insight of Bengio et al. In the paper, we also show how the original Lee-Carter model can be expressed as a neural network with embeddings! Here is a picture of the network we have just described: In the paper, we also employ one of the most interesting techniques to emerge from the computer vision literature in the past several years. The original insight is due to the authors of the ResNet paper, who analysed the well-known problem that it is often difficult to train deep neural networks. They considered that a deep neural network should be no more difficult to train than a shallow network, since the deeper layers could simply learn the identity function, and thus be equivalent to a shallow network. Without going too far off track into these details, their solution is simple â€" add skip layers that connect the deep layers to more shallow layers in the network. This idea is expanded on in the DenseNet architectures. We simply added a connection between the feature layer, and the fifth layer of the network, connecting the embedding layers almost to the deepest layer of the network. This boosted the performance of the networks considerably. We found that the deep neural networks dramatically outperformed the competing methods that we tested, forecasting with the lowest MSE in 51 out of the 76 instances we tested! Here is a table comparing the methods, and see the paper for more

details: Here is a picture of the age embedding, which shows that the main relationship learned by the network is the characteristic shape of a modern life table: This is a rather striking result, since at no time did we specify this to the network! Also, once the architecture was specified, the network has also learned to forecast mortality rates more successfully than human specified models reminding me of one of the desiderata for AI systems listed by Bengio

DOWNLOAD PDF NEURAL NETWORK MODELS

Chapter 2 : Artificial Neural Network Model - Data Science Central

Neural network models can be viewed as simple mathematical models defining a function: â†' or a distribution over or both and. Sometimes models are intimately associated with a particular learning rule.

Beginners , Neural Networks , R , Udemy In this article we will learn how Neural Networks work and how to implement them with the R programming language! We will see how we can easily create Neural Networks with R and even visualize them. Basic understanding of R is necessary to understand this article. Check out the end of the article for discount coupons on my courses! Neural Networks Neural Networks are a machine learning framework that attempts to mimic the learning pattern of natural biological neural networks. Biological neural networks have interconnected neurons with dendrites that receive inputs, then based on these inputs they produce an output signal through an axon to another neuron. We will try to mimic this process through the use of Artificial Neural Networks ANN , which we will just refer to as neural networks from now on. The process of creating a neural network begins with the most basic form, a single perceptron. A perceptron has one or more inputs, a bias, an activation function, and a single output. The perceptron receives inputs, multiplies them by some weight, and then passes them into an activation function to produce an output. There are many possible activation functions to choose from, such as the logistic function, a trigonometric function, a step function etc. We also make sure to add a bias to the perceptron, this avoids issues where all inputs could be equal to zero meaning no multiplicative weight would have an effect. Check out the diagram below for a visualization of a perceptron: Once we have the output we can compare it to a known label and adjust the weights accordingly the weights usually start off with random initialization values. We keep repeating this process until we have reached a maximum number of allowed iterations, or an acceptable error rate. To create a neural network, we simply begin to add layers of perceptrons together, creating a multi-layer perceptron model of a neural network. For a visualization of this check out the diagram below source: Rate Abilene Christian University 60 Adelphi University 56 Data Preprocessing It is important to normalize data before training a neural network on it. The neural network may have difficulty converging before the maximum number of iterations allowed if the data is not normalized. There are a lot of different methods for normalization of data. We will use the built-in scale function in R to easily accomplish this task. Usually it is better to scale the data from 0 to 1, or -1 to 1. We can specify the center and scale as additional arguments in the scale function. Undergrad Abilene Christian University 0. Board Abilene Christian University 0. Rate Abilene Christian University 0.

Chapter 3 : Data Mining Group - PMML - Neural Network Models

A neural network is a powerful computational data model that is able to capture and represent complex input/output relationships. The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain.

It maps sets of input data onto a set of appropriate outputs. Each neuron is a linear equation like linear regression as shown in the following equation Linear Regression Equations The equation is the transfer function in a neural network. This linear weight sum would be a threshold at some value so that output of neuron would be either 1 or 0. The multilayer perceptron networks are suitable for the discovery of complex nonlinear models. On the possibility of approximating any regular function with a sum of sigmoid its power based. This requires a known, desired output for each input value to calculate the loss function gradient. MLP is a modification of the standard linear perceptron and can distinguish data that are not linearly separable. But, RBF network works with only one hidden layer. It accomplishes this by calculating the value of each unit in the hidden layer for an observation. It uses the distance in space between this observation and the center of the unit. Instead of the sum of the weighted values of the units of the preceding level. Unlike the weights of a multilayer perceptron. The centers of the hidden layer of an RBF network are not adjusted at the each iteration during learning. In RBF network, hidden neurons share the space and are virtually independent of each other. This makes for faster convergence of RBF networks in the learning phase, which is one of their strong points. Response surface of a unit of the hidden layer of an RBF network is a hypersphere. The response of the unit to an individual xi is a decreasing function G of the distance between the individual and its hypersphere. The response surface of the unit, after the application of the transfer function, is a Gaussian surface. Learning of RBF involves determining the number of units in the hidden layer. Like a number of radial functions, their centers, radii, and coefficients. It is also called as self-organizing feature map SOFM. It produces a low-dimensional discretized representation of the input space of the training samples called a map. The Kohonen network is the most common unsupervised learning network. Like any neural network, it is being made up of layers of units and connections between these units. The major difference from the rest of neural networks is that there is no variable that can predict. By the following two levels the Kohonen network composed. The key related to Kohonen networks: There is also an input layer. The input layer is fully connected to the competitive layer. The units in the competitive layer sum their weighted inputs to find a single winner. It adjusts the weight of the winner and its neighbors. The adjustment is such that two close placed output units correspond to two close placed individuals. At the output Groups clusters of units forms. In the application phase, Kohonen network operates by representing each input individual by the unit of the network. The network which is closest to it in terms of a distance defined above. This unit will be the cluster of the individual.

Chapter 4 : The Neural Networks Model

Neural network models. Artificial neural networks are forecasting methods that are based on simple mathematical models of the brain. They allow complex nonlinear relationships between the response variable and its predictors.

A new convolutional neural network model to detect abuse and incivility on Twitter by Ingrid Fadelli , Tech Xplore Schematic of the steps for incivility detection. The yellow colored blocks represent inputs, the red colored blocks represent the classifiers and the blue colored blocks represent the intermediate steps. Researchers at Northwestern University, McGill University, and the Indian Institute of Technology Kharagpur have recently developed a character level convolutional neural network CNN model that could help to detect abusive posts on Twitter. This model was found to outperform several baseline methods, achieving an accuracy of In recent years, abusive behavior on online platforms has been rising exponentially, particularly on Twitter. Social media companies are hence seeking effective new methods to identify this behavior in order to intervene and prevent it from causing serious harm. In fact, this is one of the main reasons why Twitter is losing its active follower base. This is why it is important for social media platforms to detect this content effectively and rapidly, performing timely interventions to remove it. This led us to the central idea of leveraging opinion conflicts to detect uncivil tweets. They hence incorporated entity-specific sentiment information into their CNN model, hoping this would improve its performance in detecting abusive content. In the example of incivility context cited below, we observe that the target tweets positively about Donald Trump and US Economy. However, the offender account holder tweets negatively about Trump and positively about President Obama. We can observe that there is a conflict of opinion between the target and the account holder as the sentiments expressed toward the common named entity Donald Trump is opposite. Going through the entire exchange of messages, we find that this opinion conflict eventually leads to an uncivil post. As tweets are usually small, only contain a few words, and have a lot of spelling variations, character-level models are found to be more robust than word-level models. The researchers also carried out a post-hoc analysis, taking a closer look at behavioral aspects of offenders and victims on Twitter, hoping to better understand incivility incidents. This analysis revealed that a sizable portion of users were repeated offenders who had harassed targets over 10 times. Similarly, some targets had been been harassed by different offenders on a several different occasions. The researchers are now trying to develop similar models to detect hate speech on Twitter, as well as on other social media platforms. Explore further More information: An effective route to detect incivility in Twitter. A new convolutional neural network model to detect abuse and incivility on Twitter , October 3 retrieved 10 November from https: Apart from any fair dealing for the purpose of private study or research, no part may be reproduced without the written permission. The content is provided for information purposes only.

Chapter 5 : What are Neural Networks & Predictive Data Analytics?

Further, the model supports multi-label classification in which a sample can belong to more than one class. For each class, the raw output passes through the logistic function. For each class, the raw output passes through the logistic function.

I love Data Science. We asked a data scientist, Neelabh Pant, to tell you about his experience of forecasting exchange rates using recurrent neural networks. As an Indian guy living in the US, I have a constant flow of money from home to me and vice versa. If the dollar is weaker, you spend less rupees to buy the same dollar. Looking at the strengths of a neural network, especially a recurrent neural network, I came up with the idea of predicting the exchange rate between the USD and the INR. There are a lot of methods of forecasting exchange rates such as: Purchasing Power Parity PPP , which takes the inflation into account and calculates inflation differential. Relative Economic Strength Approach, which considers the economic growth of countries to predict the direction of exchange rates. Econometric model is another common technique used to forecast the exchange rates which is customizable according to the factors or attributes the forecaster thinks are important. There could be features like interest rate differential between two different countries, GDP growth rates, income growth rates, etc. Time series model is purely dependent on the idea that past behavior and price patterns can be used to predict future price behavior. Sequence problems Let us begin by talking about sequence problems. The simplest machine learning problem involving a sequence is a one to one problem. Linear regression, classification, and even image classification with convolutional network fall into this category. We can extend this formulation to allow for the model to make use of the pass values of the input and the output. It is known as the one to many problem. The one to many problem starts like the one to one problem where we have an input to the model and the model generates one output. However, the output of the model is now fed back to the model as a new input. The model now can generate a new output and we can continue like this indefinitely. You can now see why these are known as recurrent neural networks. In other words, they can retain state from one iteration to the next by using their own output as input for the next step. In programming terms this is like running a fixed program with certain inputs and some internal variables. The simplest recurrent neural network can be viewed as a fully connected neural network if we unroll the time axes. The weight multiplying the current input xt, which is u, and the weight multiplying the previous output yt-1, which is w. This formula is like the exponential weighted moving average EWMA by making its pass values of the output with the current values of the input. One can build a deep recurrent neural network by simply stacking units to one another. A simple recurrent neural network works well only for a short-term memory. We will see that it suffers from a fundamental problem if we have a longer time dependency. This is a problem because we want our RNNs to analyze text and answer questions, which involves keeping track of long sequences of words. LSTM has an internal state variable, which is passed from one cell to another and modified by Operation Gates. Forget Gate It is a sigmoid layer that takes the output at t-1 and the current input at time t and concatenates them into a single tensor and applies a linear transformation followed by a sigmoid. Because of the sigmoid, the output of this gate is between 0 and 1. This number is multiplied with the internal state and that is why the gate is called a forget gate. Input Gate The input gate takes the previous output and the new input and passes them through another sigmoid layer. This gate returns a value between 0 and 1. The value of the input gate is multiplied with the output of the candidate layer. This layer applies a hyperbolic tangent to the mix of input and previous output, returning a candidate vector to be added to the internal state. The internal state is updated with this rule: The previous state is multiplied by the forget gate and then added to the fraction of the new candidate allowed by the output gate. Output Gate This gate controls how much of the internal state is passed to the output and it works in a similar way to the other gates. These three gates described above have independent weights and biases, hence the network will learn how much of the past output to keep, how much of the current input to keep, and how much of the internal state to send out to the output. In a recurrent neural network, you not only give the network the data, but also the state of the network one moment before. It is a story where the main character is Neelabh and something happened on the road. As

you listen to all my other sentences you have to keep a bit of information from all past sentences around in order to understand the entire story. Another example is video processing, where you would again need a recurrent neural network. What happens in the current frame is heavily dependent upon what was in the last frame of the movie most of the time. Over a period of time, a recurrent neural network tries to learn what to keep and how much to keep from the past, and how much information to keep from the present state, which makes it so powerful as compared to a simple feed forward neural network. Time Series Prediction I was impressed with the strengths of a recurrent neural network and decided to use them to predict the exchange rate between the USD and the INR. The dataset used in this project is the exchange rate data between January 2, and August 10, We have a total of 13, records starting from January 2, to August 10, One can see that there was a huge dip in the American economy during â€", which was hugely caused by the great recession during that period. It was a period of general economic decline observed in world markets during the late s and early s. Many of the newer developed economies suffered far less impact, particularly China and India, whose economies grew substantially during this period. Test-Train Split Now, to train the machine we need to divide the dataset into test and training sets. It is very important when you do time series to split train and test with respect to a certain date. In our experiment, we will define a date, say January 1, , as our split date. The training data is the data between January 2, and December 31, , which are about 11, training data points. The test dataset is between January 1, and August 10, , which are about 2, points. Train-Test Split The next thing to do is normalize the dataset. You only need to fit and transform your training data and just transform your test data. Normalizing or transforming the data means that the new scale variables will be between zero and one. This basically takes the price from the previous day and forecasts the price of the next day. As a loss function, we use mean squared error and stochastic gradient descent as an optimizer, which after enough numbers of epochs will try to look for a good local optimum. Below is the summary of the fully connected layer. Since we split the data into training and testing sets we can now predict the value of testing data and compare them with the ground truth. Ground Truth blue vs Prediction orange As you can see, the model is not good. It essentially is repeating the previous values and there is a slight shift. The fully connected model is not able to predict the future from the single previous value. Let us now try using a recurrent neural network and see how well it does. Long Short-Term Memory The recurrent model we have used is a one layer sequential model. We used 6 LSTM nodes in the layer to which we gave input of shape 1,1 , which is one input given to the network with one value. The summary of the model is shown above. It is still underestimating some observations by certain amounts and there is definitely room for improvement in this model. One can always try to change the configuration by changing the optimizer. Another important change I see is by using the Sliding Time Window method, which comes from the field of stream data management system. This approach comes from the idea that only the most recent data are important. One can show the model data from a year and try to make a prediction for the first day of the next year. Sliding time window methods are very useful in terms of fetching important patterns in the dataset that are highly dependent on the past bulk of observations. Try to make changes to this model as you like and see how the model reacts to those changes. Dataset I made the dataset available on my github account under deep learning in python repository. Feel free to download the dataset and play with it. Try to keep up with the news of different artificial intelligence conferences. Conclusion LSTM models are powerful enough to learn the most important past behaviors and understand whether or not those past behaviors are important features in making future predictions. There are several applications where LSTMs are highly used. Applications like speech recognition, music composition, handwriting recognition, and even in my current research of human mobility and travel predictions. According to me, LSTM is like a model which has its own memory and which can behave like an intelligent human in making decisions. Thank you again and happy machine learning!

Chapter 6 : What is an Artificial Neural Network (ANN)? - Definition from Techopedia

Neural Network Models for Backpropagation The description of neural network models assumes that the reader has a general knowledge of artificial neural network technology. A neural network has one or more input nodes and one or more neurons.

Neural Network Definition Neural networks are a set of algorithms, modeled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labeling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated. Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabeled data according to similarities among the example inputs, and they classify data when they have a labeled dataset to train on. Neural networks can also extract features that are fed to other algorithms for clustering and classification; so you can think of deep neural networks as components of larger machine-learning applications involving algorithms for reinforcement learning , classification and regression. What kind of problems does deep learning solve, and more importantly, can it solve yours? To know the answer, you need to ask questions: What outcomes do I care about? Those outcomes are labels that could be applied to data: Do I have the data to accompany those labels? A Few Concrete Examples Deep learning maps inputs to outputs. Here are a few examples of what deep learning can do. Classification All classification tasks depend upon labeled datasets; that is, humans must transfer their knowledge to the dataset in order for a neural to learn the correlation between labels and data. This is known as supervised learning. Detect faces, identify people in images, recognize facial expressions angry, joyful Identify objects in images stop signs, pedestrians, lane markersâ€¦ Recognize gestures in video Detect voices, identify speakers, transcribe speech to text, recognize sentiment in voices Classify text as spam in emails , or fraudulent in insurance claims ; recognize sentiment in text customer feedback Any labels that humans can generate, any outcomes you care about and which correlate to data, can be used to train a neural network. Clustering Clustering or grouping is the detection of similarities. Deep learning does not require labels to detect similarities. Learning without labels is called unsupervised learning. Unlabeled data is the majority of data in the world. One law of machine learning is: Therefore, unsupervised learning has the potential to produce highly accurate models. Comparing documents, images or sounds to surface similar items. The flipside of detecting similarities is detecting anomalies, or unusual behavior. In many cases, unusual behavior correlates highly with things you want to detect and prevent, such as fraud. Regressions With classification, deep learning is able to establish correlations between, say, pixels in an image and the name of a person. You might call this a static prediction. By the same token, exposed to enough of the right data, deep learning is able to establish correlations between present events and future events. It can run regression between the past and the future. The future event is like the label in a sense. Given a time series, deep learning may read a string of number and predict the number most likely to occur next. Hardware breakdowns data centers, manufacturing, transport Health breakdowns strokes, heart attacks based on vital stats and data from wearables Customer churn predicting the likelihood that a customer will leave, based on web activity and metadata Employee turnover ditto, but for employees The better we can predict, the better we can prevent and pre-empt. Not zero surprises, just marginally fewer. The layers are made of nodes. A node is just a place where computation happens, loosely patterned on a neuron in the human brain, which fires when it encounters sufficient stimuli. A node combines input from the data with a set of coefficients, or weights, that either amplify or dampen that input, thereby assigning significance to inputs for the task the algorithm is trying to learn. For example, which input is most helpful is classifying data without error? A node layer is a row of those neuronlike switches that turn on or off as the input is fed through the net. Pairing adjustable weights with input features is how we assign significance to those features with regard to how the network classifies and clusters input. Key Concepts of Deep Neural Networks Deep-learning networks are distinguished from the more commonplace single-hidden-layer neural networks by their depth; that is, the number of node layers through which data

passes in a multistep process of pattern recognition. Earlier versions of neural networks such as the first perceptrons were shallow, composed of one input and one output layer, and at most one hidden layer in between. So deep is a strictly defined, technical term that means more than one hidden layer. The further you advance into the neural net, the more complex the features your nodes can recognize, since they aggregate and recombine features from the previous layer. This is known as feature hierarchy, and it is a hierarchy of increasing complexity and abstraction. It makes deep-learning networks capable of handling very large, high-dimensional data sets with billions of parameters that pass through nonlinear functions. Above all, these nets are capable of discovering latent structures within unlabeled, unstructured data, which is the vast majority of data in the world. Another word for unstructured data is raw media; i. For example, deep learning can take a million images, and cluster them according to their similarities: This is the basis of so-called smart photo albums. Now apply that same idea to other data types: Deep learning might cluster raw text such as emails or news articles. Emails full of angry complaints might cluster in one corner of the vector space, while satisfied customers, or spambot messages, might cluster in others. This is the basis of various messaging filters, and can be used in customer-relationship management CRM. The same applies to voice messages. Deep-learning networks perform automatic feature extraction without human intervention, unlike most traditional machine-learning algorithms. Given that feature extraction is a task that can take teams of data scientists years to accomplish, deep learning is a way to circumvent the chokepoint of limited experts. It augments the powers of small data science teams, which by their nature do not scale. Restricted Boltzmann machines, for examples, create so-called reconstructions in this manner. In the process, these networks learn to recognize correlations between certain relevant features and optimal results â€" they draw connections between feature signals and what those features represent, whether it be a full reconstruction, or with labeled data. A deep-learning network trained on labeled data can then be applied to unstructured data, giving it access to much more input than machine-learning nets. This is a recipe for higher performance: Bad algorithms trained on lots of data can outperform good algorithms trained on very little. Deep-learning networks end in an output layer: We call that predictive, but it is predictive in a broad sense. Given raw data in the form of an image, a deep-learning network may decide, for example, that the input data is 90 percent likely to represent a person. Feedforward Networks Our goal in using a neural net is to arrive at the point of least error as fast as possible. We are running a race, and the race is around a track, so we pass the same points repeatedly in a loop. The starting line for the race is the state in which our weights are initialized, and the finish line is the state of those parameters when they are capable of producing accurate classifications and predictions. The race itself involves many steps, and each of those steps resembles the steps before and after. Just like a runner, we will engage in a repetitive act over and over to arrive at the finish. Each step for a neural network involves a guess, an error measurement and a slight update in its weights, an incremental adjustment to the coefficients. Models normally start out bad and end up less bad, changing over time as the neural network updates its parameters. This is because a neural network is born in ignorance. It does not know which weights and biases will translate the input best to make the correct guesses. It has to start out with a guess, and then try to make better guesses sequentially as it learns from its mistakes. You can think of a neural network as a miniature enactment of the scientific method, testing hypotheses and trying again â€" only it is the scientific method with a blindfold on. Here is a simple explanation of what happens during learning with a feedforward neural network, the simplest architecture to explain. Input enters the network. The coefficients, or weights, map that input to a set of guesses the network makes at the end. The network measures that error, and walks the error back over its model, adjusting weights to the extent that they contributed to the error. A neural network is a corrective feedback loop, rewarding weights that support its correct guesses, and punishing weights that lead it to err. Multiple Linear Regression Despite their biologically inspired name, artificial neural networks are nothing more than math and code, like any other machine-learning algorithm. In fact, anyone who understands linear regression, one of first methods you learn in statistics, can understand how a neural net works. To make this more concrete: X could be radiation exposure and Y could be the cancer risk; X could be daily pushups and Y could be the total weight you can benchpress; X the amount of fertilizer and Y the size of the crop. You can imagine that every time you add a unit to X, the dependent variable Y increases proportionally, no matter how

far along you are on the X axis. That simple relation between two variables moving up or down together is a starting point. The next step is to imagine multiple linear regression, where you have many input variables producing an output variable. Now, that form of multiple linear regression is happening at every node of a neural network. For each node of a single layer, input from each node of the previous layer is recombined with input from every other node. That is, the inputs are mixed in different proportions, according to their coefficients, which are different leading into each node of the subsequent layer. In this way, a net tests which combination of input is significant as it tries to reduce error. What we are trying to build at each node is a switch like a neuronâ€¦ that turns on and off, depending on whether or not it should let the signal of the input pass through to affect the ultimate decisions of the network. When you have a switch, you have a classification problem. A binary decision can be expressed by 1 and 0, and logistic regression is a non-linear function that squashes input to translate it to a space between 0 and 1. The nonlinear transforms at each node are usually s-shaped functions similar to logistic regression. The output of all nodes, each squashed into an s-shaped space between 0 and 1, is then passed as input to the next layer in a feed forward neural network, and so on until the signal reaches the final layer of the net, where decisions are made. To put a finer point on it, which weight will produce the least error?

Chapter 7 : A Beginner's Guide to Neural Networks and Deep Learning | Skymind

Convolutional Neural Networks (CNNs) is the most popular neural network model being used for image classification problem. The big idea behind CNNs is that a local understanding of an image is good enough.

Deep learning is a class of machine learning algorithms that: Each successive layer uses the output from the previous layer as input. Overview[edit] Most modern deep learning models are based on an artificial neural network , although they can also include propositional formulas or latent variables organized layer-wise in deep generative models such as the nodes in deep belief networks and deep Boltzmann machines. In an image recognition application, the raw input may be a matrix of pixels; the first representational layer may abstract the pixels and encode edges; the second layer may compose and encode arrangements of edges; the third layer may encode a nose and eyes; and the fourth layer may recognize that the image contains a face. Importantly, a deep learning process can learn which features to optimally place in which level on its own. Of course, this does not completely obviate the need for hand-tuning; for example, varying numbers of layers and layer sizes can provide different degrees of abstraction. More precisely, deep learning systems have a substantial credit assignment path CAP depth. The CAP is the chain of transformations from input to output. CAPs describe potentially causal connections between input and output. For a feedforward neural network , the depth of the CAPs is that of the network and is the number of hidden layers plus one as the output layer is also parameterized. For recurrent neural networks , in which a signal may propagate through a layer more than once, the CAP depth is potentially unlimited. CAP of depth 2 has been shown to be a universal approximator in the sense that it can emulate any function. Deep learning architectures are often constructed with a greedy layer-by-layer method. Deep learning algorithms can be applied to unsupervised learning tasks. This is an important benefit because unlabeled data are more abundant than labeled data. Examples of deep structures that can be trained in an unsupervised manner are neural history compressors [13] and deep belief networks. It features inference, [10] [11] [1] [2] [14] [20] as well as the optimization concepts of training and testing , related to fitting and generalization , respectively. More specifically, the probabilistic interpretation considers the activation nonlinearity as a cumulative distribution function. While the algorithm worked, training required 3 days. Because it directly used natural images, Cresceptron started the beginning of general-purpose visual learning for natural 3D worlds. Cresceptron is a cascade of layers similar to Neocognitron. But while Neocognitron required a human programmer to hand-merge features, Cresceptron learned an open number of features in each layer without supervision, where each feature is represented by a convolution kernel. Cresceptron segmented each learned object from a cluttered scene through back-analysis through the network. Max pooling , now often adopted by deep neural networks e. ImageNet tests , was first used in Cresceptron to reduce the position resolution by a factor of 2x2 to 1 through the cascade for better generalization. Each layer in the feature extraction module extracted features with growing complexity regarding the previous layer. Both shallow and deep learning e. Most speech recognition researchers moved away from neural nets to pursue generative modeling. An exception was at SRI International in the late s. The principle of elevating "raw" features over hand-crafted optimization was first explored successfully in the architecture of deep autoencoder on the "raw" spectrogram or linear filter-bank features in the late s, [48] showing its superiority over the Mel-Cepstral features that contain stages of fixed transformation from spectrograms. The raw features of speech, waveforms , later produced excellent larger-scale results. In , LSTM started to become competitive with traditional speech recognizers on certain tasks. Deep learning is part of state-of-the-art systems in various disciplines, particularly computer vision and automatic speech recognition ASR. The NIPS Workshop on Deep Learning for Speech Recognition [68] was motivated by the limitations of deep generative models of speech, and the possibility that given more capable hardware and large-scale data sets that deep neural nets DNN might become practical. It was believed that pre-training DNNs using generative models of deep belief nets DBN would overcome the main difficulties of neural nets. DNN models, stimulated early industrial investment in deep learning for speech recognition, [71] [68] eventually leading to pervasive and dominant use in that industry. That analysis was done with comparable performance less than 1. While there, Ng determined

that GPUs could increase the speed of deep-learning systems by about times. In October , a similar system by Krizhevsky et al. In November , Ciresan et al. The Wolfram Image Identification project publicized these improvements.

Chapter 8 : A Guide For Time Series Prediction Using Recurrent Neural Networks (LSTMs)

The Keras Python deep learning library provides tools to visualize and better understand your neural network models. In this tutorial, you will discover exactly how to summarize and visualize your deep learning models in Keras.

September 17, 1. Objective In this Machine Learning tutorial, we will take you through the introduction of Artificial Neural network Model. First of all, we will discuss the multilayer Perceptron network next with the Radial Basis Function Network, they both are supervised learning model. At last, we will cover the Kohonen Model which follows Unsupervised learning and the difference between the Multilayer Perceptron network and Radial Basis Function Network. Introduction to Artificial Neural Network Model 2. Main ones are Multilayer Perceptron â€" It is a feedforward artificial neural network model. It maps sets of input data onto a set of appropriate outputs. Radial Basis Function Network â€" A radial basis function network is an artificial neural network. It uses radial basis functions as activation functions. Both of the above are being supervised learning networks used with 1 or more dependent variables at the output. The Kohonen Network â€" It is an unsupervised learning network used for clustering. Multilayer Perceptron As we saw above, A multilayer perceptron is a feedforward artificial neural network model. An MLP consists of many layers of nodes in a directed graph, with each layer connected to the next one. Each neuron is a linear equation like linear regression as shown in the following equation The equation is the transfer function in a neural network. This linear weight sum would be a threshold at some value so that output of neuron would be either 1 or 0. The multilayer perceptron networks are suitable for the discovery of complex nonlinear models. On the possibility of approximating any regular function with a sum of sigmoid its power based. MLP utilizes a supervised learning technique called backpropagation for training the network. This requires a known, desired output for each input value to calculate the loss function gradient. MLP is a modification of the standard linear perceptron and can distinguish data that are not linearly separable. Do you know about ANN Applications 2. But, RBF network works with only one hidden layer. It accomplishes this by calculating the value of each unit in the hidden layer for an observation. It uses the distance in space between this observation and the center of the unit. Instead of the sum of the weighted values of the units of the preceding level. Unlike the weights of a multilayer perceptron. The centers of the hidden layer of an RBF network are not adjusted at each iteration during learning. In RBF network, hidden neurons share the space and are virtually independent of each other. This makes for faster convergence of RBF networks in the learning phase, which is one of their strong points. The response of the unit to an individual xi is a decreasing function G of the distance between the individual and its hypersphere. The response surface of the unit, after the application of the transfer function, is a Gaussian surface. Learning of RBF involves determining the number of units in the hidden layer. Like a number of radial functions, their centers, radii, and coefficients. It is also called as self-organizing feature map SOFM. It produces a low-dimensional discretized representation of the input space of the training samples called a map. The Kohonen network is the most common unsupervised learning network. Like any neural network, it is being made up of layers of units and connections between these units. The major difference from the rest of neural networks is that there is no variable that can predict. By the following two levels the Kohonen network composed. The key related to Kohonen networks: There is also an input layer. The input layer is fully connected to the competitive layer. The units in the competitive layer sum their weighted inputs to find a single winner. It adjusts the weight of the winner and its neighbors. The adjustment is such that two close placed output units correspond to two close placed individuals. At the output Groups clusters of units forms. Do you know about Kernel Functions In the application phase, Kohonen network operates by representing each input individual by the unit of the network. The network which is closest to it in terms of a distance defined above. This unit will be the cluster of the individual. They also provide very little insight into what these model do. Also, neural network users must make many modeling assumptions. For example, the number of hidden layers and the number of units in each hidden layer, and usually, there is little guidance on how to do this. Thus it takes the considerable experience to determine the most appropriate representation. Furthermore, back-propagation can be quite slow if the learning constant is not in the correct form. They have

much more in common than most of the NN literature would suggest. The only difference is the way in which hidden units combine values coming from preceding layers in the network. An MLP has one or more hidden layers for which the combination function is the inner product of the inputs and weights, plus a bias. The activation function is usually a logistic or tanh function. MLP provides better generalization as it has the number of hidden units while RBF has less risk of non-optimal convergence. RBF networks usually have only one hidden layer. By which the combination function depends on the Euclidean distance between the input vector and the weight vector. Instead of adding it in the combination function like a bias, you divide the Euclidean distance by the width.

Chapter 9 : Artificial Neural Network Model Essay

A Basic Introduction To Neural Networks What Is A Neural Network? The simplest definition of a neural network, more properly referred to as an 'artificial' neural network (ANN), is provided by the inventor of one of the first neurocomputers, Dr. Robert Hecht-Nielsen.

The motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain. Neural networks resemble the human brain in the following two ways: A neural network acquires knowledge through learning. The true power and advantage of neural networks lies in their ability to represent both linear and non-linear relationships and in their ability to learn these relationships directly from the data being modeled. Traditional linear models are simply inadequate when it comes to modeling data that contains non-linear characteristics. This type of neural network is known as a supervised network because it requires a desired output in order to learn. The goal of this type of network is to create a model that correctly maps the input to the output using historical data so that the model can then be used to produce the output when the desired output is unknown. A graphical representation of an MLP is shown below: The inputs are fed into the input layer and get multiplied by interconnection weights as they are passed from the input layer to the first hidden layer. Within the first hidden layer, they get summed then processed by a nonlinear function usually the hyperbolic tangent. As the processed data leaves the first hidden layer, again it gets multiplied by interconnection weights, then summed and processed by the second hidden layer. Finally the data is multiplied by interconnection weights then processed one last time within the output layer to produce the neural network output. The MLP and many other neural networks learn using an algorithm called backpropagation. With backpropagation, the input data is repeatedly presented to the neural network. With each presentation the output of the neural network is compared to the desired output and an error is computed. This error is then fed back backpropagated to the neural network and used to adjust the weights such that the error decreases with each iteration and the neural model gets closer and closer to producing the desired output. This process is known as "training". The demonstration of a neural network learning to model using the exclusive-or Xor data. The Xor data is repeatedly presented to the neural network. With each presentation, the error between the network output and the desired output is computed and fed back to the neural network. The neural network uses this error to adjust its weights such that the error will be decreased. This sequence of events is usually repeated until an acceptable error has been reached or until the network no longer appears to be learning. A good way to introduce the topic is to take a look at a typical application of neural networks. OCR software allows you to scan in a printed document and then convert the scanned image into to an electronic text format such as a Word document, enabling you to manipulate the text. Some of the OCR software on the market use a neural network as the classification engine. The demonstration of a neural network used within an optical character recognition OCR application. The original document is scanned into the computer and saved as an image. The OCR software breaks the image into sub-images, each containing a single character. The sub-images are then translated from an image format into a binary format, where each 0 and 1 represents an individual pixel of the sub-image. The binary data is then fed into a neural network that has been trained to make the association between the character image data and a numeric value that corresponds to the character. Of course character recognition is not the only problem that neural networks can solve. Neural networks have been successfully applied to broad spectrum of data-intensive applications, such as: Process Modeling and Control - Creating a neural network model for a physical plant then using that model to determine the best control settings for the plant. Machine Diagnostics - Detect when a machine has failed so that the system can automatically shut down the machine when this occurs. Portfolio Management - Allocate the assets in a portfolio in a way that maximizes return and minimizes risk. Targeted Marketing - Finding the set of demographics which have the highest response rate for a particular marketing campaign. Financial Forecasting - Using the historical data of a security to predict the future movement of that security. Quality Control - Attaching a camera or sensor to the end of a production process to automatically inspect for defects. Fraud

Detection - Detect fraudulent credit card transactions and automatically decline the charge. Looking for neural network software to solve your own problem? NeuroDimension has been in the business of bringing neural networks and predictive data analytics to individuals, businesses, and universities from around the world for over 20 years now. Our NeuroSolutions software is a leader in allowing researchers to apply both classic and custom neural networks to their data. And, based on this experience, we are happy to bring you the next generation of predictive data analytics in NeuroSolutions Infinity. NeuroSolutions Infinity performs all of the leg work of predictive data analytics for you. It accesses your data, cleans it, organizes it, manipulates it, and intelligently searches through the most popular neural networks. It also features next generation distributed and parallel computing using as many computers and processors as you want at discovering relationships. And, it even lets you use your solution directly from within your own custom software with the Infinity QuickDeploy add-on. To download a free trial of NeuroSolutions Infinity please visit our Download page. All trademarks and copyrights are the property of their respective owners.