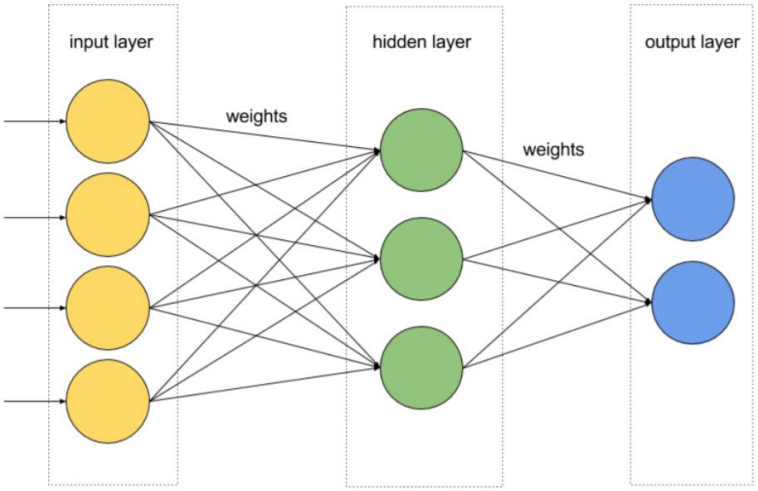
NEURAL NETWORKS

A neural network can be defined as a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs.Moreover, Artifical networks can be thought as computational model that is inspired by the way biological neural networks in the human brain process information.

Artificial neural networks are made up of artificial neurons(called also nodes or units), where each neuron is a mathematical function. It receives input from some other nodes, or from an external source and computes an output.

An artificial neural network consists of several layers of neurons – one input layer, one or more hidden layers and one output layer. The neurons are connected between each layer, and their connections are weighted with a numerical value. There are several types of artificial neural networks, with feedforward networks being the simplest form. Information is never fed backwards through the feedforward network, it is always moving forward. This is done by using the output from the preceding layer as input for the current.



Each of the hidden neurons have a weighted input from the previous layer. The weights are a measure of how much importance the neuron places upon the given input. The neuron may also have a bias which decide the firing rate of the given neuron, which is how likely the neuron is to give a high output value. This is then fed through an activation function, and sent to the next layer [3]. This can be written mathematically as: $y = f(x) = \sigma(xW + b)$ where x is the vector input, σ is the activation function, b is the bias, W is the weight and y is the output of the neuron. As the input is multiplied by the weight, an activation function is used to keep the output of the neuron bounded. An often-used activation function is the sigmoid function which "squashes" the calculated output of the neuron [4], to ensure that the value stays between 0 and 1:

$\sigma(z) = 1\ 1 + e - z$ Other activation functions are the hyperbolic tangent [4], which keeps the value between -1 and 1, as well as the rectified linear unit [5]: $\sigma(z) = \max(0, z)$

A neural network is deemed to be deep when it consists of more than one hidden layer. The neurons of each layer specialise in learning a certain feature.

In deep networks there might be a about feature hierarchy, where the lower-end layers recognize simple features and higher-end layers recognizing more complex and abstract features.

e.g: face recognition.

BACKWARD PROPAGATION

Basically,Backward Prorogation of Errors, often abbreviated as BackProp is one of the several ways in which an artificial neural network (ANN) can be trained. It is a supervised training scheme, which means, it learns from labeled training data (there is a supervisor, to guide its learning).
To put in simple terms, BackProp is like "learning from mistakes". The supervisor he ANN whenever it makes mistakes. Below we describe some details of

When training a neural network, the network is presented with a set of training data (P) containing both input data (x) and its corresponding output results (y): $P(x, y) = (x0, y0\ ), (x1, y1\ ), \dots , (xi\ , yi)$ The goal of the training is to find the best set of weights and biases for all the connections and neurons. First, the input data is fed forward through the neural network (N), generating output y': $x \rightarrow N \rightarrow y'$.

Then the error of the output is calculated by taking the difference between each pair of output y' and y, multiplied by the derivative of their activation function.

The second step is to compute the error backwards through each layer to adjust the weights. If the neural network is viewed as a composite function made up of inputs, outputs and the hidden layers, we can then add the cost function E(X) at the end as a function of the output [7]. The hidden neurons can be written as h1 (x), … hn (x) with output O(x), giving the cost function $E(x) = O\ (hn(\dots\ (h1(x))))$. Using the chain rule on the cost function, the error is propagated backwards for each layer, meaning backpropagation calculates the slope $\partial E\ \partial W$ of each step to get the derivative of the full chain

DIFFERENCE BETWEEN REGULAR FEEDFORWARD NETWORKS AND RECURRENT NETWORKS

Regular feedforward networks take the input x, processes it forward through its neurons and produces a prediction y. As such, the data that has been fed through the network before does not affect the current input, and the processing does not change according to the surrounding context.

On the contrary ,Recurrent neural networks (RNN) that can process a sequence of vectors and let the past input affect the current, making it capable of temporal memory and decision making based on previous results.

RECURRENT NETWORKS

RNN's can store information about what was previously fed to the network, and uses this to make predictions . Mathematically this can be written as $r = \sigma(Wxt + Uht{-}1)$ As with feedforward neural networks, we have the activation function σ with current input xt, multiplied by a weight matrix W, but additionally we also have the previous output ht-1 multiplied by a transition matrix U. These matrices are used to assign the importance to the current and past input, and backpropagation is used to minimize the error caused by the weights .

A bidirectional recurrent neural network (BRNN) is a neural network that is capable of learning temporal dependencies, not only considering past input, but also future input . This is achieved by using two separate RNN's. The two RNN's are going in opposite directions – one reading the input beginning to end, while the other is reading it backwards. This way, one RNN learns to make predictions based on the past, while the other learns to make predictions based on the future. The output of these two are then combined to make a joint prediction, looking at the sequence input as a whole .A BRNN is therefore useful when the output depends not only on previous output but also on the output to come, for instance when translating sentences where the meaning of words can change based on the context.

Recurrent neural networks differ from regular feedforward networks in having the added element of time, that is, all the inputs from previous time-steps. When training a neural network with time steps, this must be considered due to the added complexity. Backpropagation through time (BPTT) is based on the regular backpropagation algorithm, except that all the time steps also are added to the nested chain function. According to Rumelhart et., this can be done by unfolding the recurrent neural network with regards to time, making it similar to a feedforward neural network. When the RNN unfolded, the network computes the error backwards through each of the layers, for every time step. When this is done, the neural network is folded back to a recurrent network, and the weight updated.

With many time steps, BPTT can be time consuming. With n number of units, it has a worst case complexity of $\Theta(n\ 2\ )$ per time-step.

With gradient-based methods such as the backpropagation algorithm, a problem with limitations in the RNN's "memory" occur., as the data passes through the neural network, it is put it though several stages of multiplication and activation functions consecutively over many time-steps. This makes the gradient prone to vanishing or exploding.

A solution to the exploding- and vanishing gradient problems is called Long short-term memory

The LSTM is a memory block that consists of memory cells and gate units, which respectively store information and control the flow of information. It has a complexity of O(W) per time step, with W being total number of weights.

LSTM avoids vanishing gradients by having two flows of data through the network; the input to output flow and the memory flow. It does not use activation functions or multiplication to alter the data stored in the memory, and as such it avoids the memory vanishing or exploding. A

LSTM network therefore has a constant error flow, with gates controlling the access to the memory cells.

CONVOLUTION NEURAL NETWORKS

Basically , Convolutional neural networks (CNN) are a group of deep feedforward neural networks, where the neurons in the hidden layers are arranged in three dimensions – height, width and depth. due to their ability to process multi-dimensional input. CNN's achieve this by mixing different types of layers, most often used are convolution, pooling and fully connected layers. Convolution layer filters the input by separating the input into smaller sub matrices (clusters of neurons) and performs dot multiplication on each of these clusters with each cluster's corresponding weights. The output of each calculation is stacked creating a three dimensional output – e.g. a [3 × 3] input with 4 filters would produce an output of [3 × 3 × 4].

Pooling layers are used to reduce the total number of neurons. This is done by looking at smaller clusters of neurons at a time, with a given size and stride. Taking the example of a [9x9] input with size [3x3] and stride 3, the pooling layer will produce a [3x3] output. Example of different pooling layers are max pooling and average pooling. An average pooling layer will output the average of each neuron cluster, and max pooling will output the highest value of each cluster

SPEECH RECOGNITION ARCHITECTURES

When classifying speech using neural networks, there are several considerations to make. Architecture wise, speech recognition has traditionally been split into several different components due to the complex nature of speech recognition. Recent efforts have gone toward making purely neural network based solutions for speech recognition, often called end-to-end networks, that can be trained as a whole network, rather than training individual components separately. For a speech recognition system using LSTM, the RNN will need input and output to be standardized. Even though the RNN can process a sequence to a sequence , the alignment between input and output data is a problem, particularly in speech recognition. It may be useful to use encoders and decoders The encoders and decoders were in this case two separate RNN's that first take the input and turn it into a fixed size vector for the RNN to process. After the input is processed, the decoder turns the fixed-size result into a sequence

HMM WITH NEURAL NETWORK

Many existing speech recognition solutions use a Hidden Markov Model (HMM) in combination with a neural network as an acoustic model .The HMM models the sequential nature of the data and the neural network estimates the observation probabilities .The model takes pre-processed sound as input, and outputs phonemes. The phonemes are searched for in a pronunciation dictionary, that find the corresponding word. A language model is then applied, which is a file containing word combinations with their statistical probability of occurring

The HMM would often be pre-trained with Gaussian mixture models before the neural network can be trained.

As the neural network would typically be trained as a classifier of the frame-level acoustic data, it necessitates a training target for each frame. This again requires the audio sequences to be aligned with the transcriptions by the HMM, but this alignment is dependent on the classification of frames. This creates a circular dependency between segmentation and recognition. The training becomes an iterative process where it is necessary to realign the HMM-network as the weights of the neural network is updated. The objective function used to train the networks is also different than the actual performance measure, the sound-sequence to text-sequence accuracy. Both introduced the attention based encoder-decoder network to speech recognition. It is a sequence to sequence network with a "listener" that encodes speech signals to high-end features, and an "speller" that decodes these features into output utterances. Chan et al proposed a model they named Listen, attend and spell (LAS), with a pyramid BLSTM as the listener, that encodes a long input sequence x into a shorter, high-end sequence h. This is then processed by the speller, which is an attention-based LSTM transducer that produces a probability distribution over character sequences . The attention-mechanism can tend to produce extremely non-sequential mappings due to its flexibility, and while audio to text mapping can be slightly non-sequential is some cases, it is still mostly a sequential mapping. This flexibility can slow down training. In addition, the varying length of audio sequences to text can cause trouble

The Microsoft conversational speech recognition system ,rather than using a single type of neural network as the acoustic model, they let several different networks process the data in parallel, and finally combine the output predictions of the different networks in an additional network.

The 2017 system uses several different networks; the CNN-variants called Res Net and LACE, a BLSTM network and a CNN-BLSTM combination. The output of these networks is fed to a LSTM-language that outputs word probabilities based on previous words. By using several different network structures, they exploit the fact that the different networks make different errors in their predictions, and as such the final combined prediction is improved. Their combined system achieved a word error rate as low as 5.1% on the Switchboard dataset.

CONSIDERATIONS

As speech can be slow or fast, high or low pitched, and different dialects and accents exist within the same language, speech recognition is a complex task, even before considering the technical aspects such as sound sampling rate, noise pollution or echo. What is often used in a lot of speech recognition services is HMM in combination with a neural network. However, the HMM based design has over the past recent years started to become less frequent in newer projects, often replaced by different solutions using end-to-end RNN with LSTM due to its ability to model long temporal dependencies. Deciding which composition or technique would be best to use, we first and foremost must bear the main goal of the system in mind; a small-scale speech recognition service specialised in recognising a few, pre-defined words or sentences would have

other system requirements than a large-scale system aiming to identify full sentences from an entire language. Furthermore, the different architectures have varying degrees of adaptability and reusability; some require expert knowledge of both the system and the language, and are difficult to adapt to new languages or changes in the system. There is also the need to consider where the system should be used, for instance are mobile devices or tablets relevant devices for this purpose, however they often come with limited storage space.

CONCLUSIONS

To answer our question "how can neural networks be used for speech recognition", we have seen that there are several solutions that potentially could be used to create an efficient solution. HMM-hybrids still dominate, but they are hard to implement, adapt and maintain, as well as having larger file sizes. Taking this into consideration as well as the aim being to investigate how to create a large-scale speech recognition system, using a purely neural network system based on LSTM seems to be a promising solution. The two main approaches involving end-to-end neural networks, the CTC-model and the attention-based encoder-decoder, are both potentially good solutions. Although these networks are often resource intensive to train, recent advances in the field with both the CTC- and the LAS-models have shown promising results. However, in order to decide between those two, we would require further investigation in future work. As a result, we would like to present two systems based on these two approaches as a conclusion.

For the CTC-model, we would recommend a BLSTM network, possibly combined with a CNN. The LSTM structure seems to be a good choice due to its ability to model temporal dependencies. We believe the addition of the CNN would help to pre-process the sound, making the features clearer for the BLSTM network. Due to limitations in CTC-model's ability to learn how to spell, the model could also benefit from having a language model processing the final results.

For the attention-based model, two neural networks each for the encoder-decoder are needed. In this case we believe the original model presented as a BLSTM-attend-LSTM is a good solution, especially with the recent improvements to the model with regards to word-piece models and Second-Pass rescoring, among others.