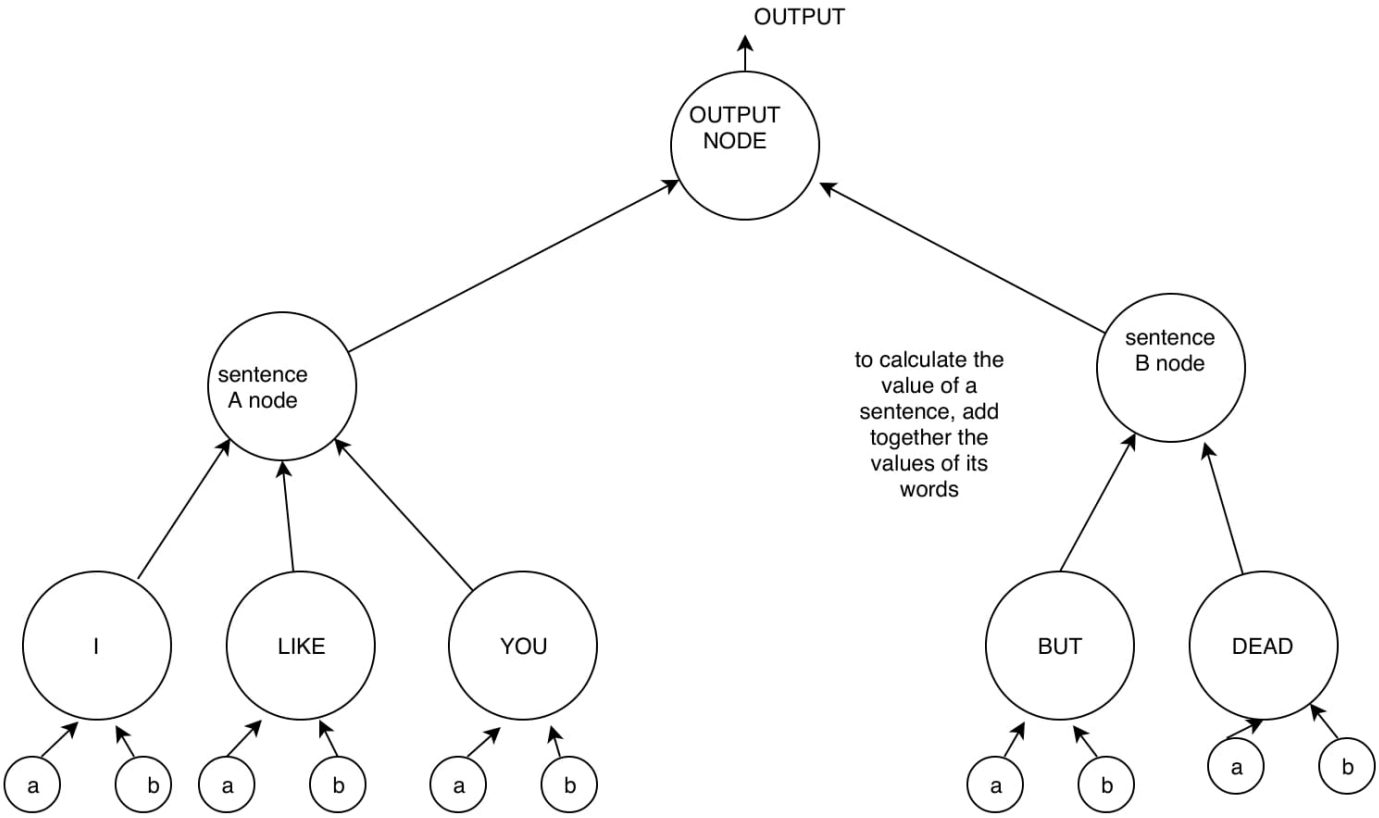
IDEALISTIC NEURAL NETWORKS

Author: Tofara Moyo

Organization: Mazusa Email: tofaramoyo@gmail.com

Typically the method of choice to represent words is by vectoring them, and placing them in some high Dimensional space. We shall not depart from this. Where we do differ is in specifying that a word has a Consistent vector representation throughout all possible sentences that it may be found in. To illustrate The new method of representation we may start by representing words with scalars for clarity. Fig 1.1 illustrates a network made of two sentences. The words in each sentence form nodes, then all The nodes in each sentence are linked to a single node, one for each sentence and finally the output Node connects the two sentence node. Note the decidedly antisocial nature of the phrases. That is Because we are setting up a network, or a system that will recognize hate speech and differentiate it From more normal speech. Below each word in the above diagram there are two nodes named “a” and “b”. These are to be seen as Indexed by the word they relate to. These variables are for this example scalars, and in further examples will be vectored. To perform a forward pass through this network we would take all the “b” variables in a particular sentence and add them together. Then, in the process of establishing a value for itself, each word in the sentence would take this sum, and multiply it by its own “a” value. This s the process that defines the indeterminacy part of the name of the network. Note that with this process, a single word can have multiple different values that depend on the values of the other words in the sentence and only once we have collected all the “b”s , once we have a sentence at hand, can we generate a value for the word. This gives us all the power of a system that maps the power set of the English language upon itself, without the explosion in the number of variables that this mapping would produce. Once a value for each word in a sentence is ascertained, the values of the words are summed and this sum becomes the value of the sentence node. To continue the forward pass the values of the sentences are summed and passed to the output node, which may be given an activation function such as a sigmoid activation function. This simplicity in the specification of the network aids greatly in maintaining the convexity of the error function, and in the Meta explanation ahead, aids in visualizing how this network actually performs. We may wish to have an output of 1 signify hate speech, while one of -1 signify normal speech. So we would perform supervised style training with labeled examples of each, back propagating the error throughout the network, ultimately reaching all the way back to the “a”’s and “b”’s of the underlying words. Note that the topology of the network will change for each example, making it a type of virtual neural network. To visualize how this network will perform during operation imagine that each word exists as a cloud in some space. Then we could draw a decision hyper plane arbitrarily in this space. As explained each word resolves to a point, this cloud, once in a sentence and we can imagine the sentence as being a hyper plane in this space. Before training the sentence hyper planes exist randomly on either side of the decision hyperplane. But after training these sentence hyper planes would have been shifted over to the relevant side of the decision boundary, by programming this asymmetrically into the

nature of the clouds of each word. So the clouds still exist at either side of the boundary, but the rule for linking points in one cloud to another, leave hate speech on one side of the decision boundary while normal speech is positioned on the other. Of course this would need to be developed to make this model more than a bag of words model, by scaling each words value by the position it occupies in a sentence, and possibly scaling sentences by their position within a passage. Typically the training will take the form of treating all the “a”’s in the network as constants and learning values for the “b”’s, then , when there is no more improvement in the accuracy of the network, we hold the learnt values of the “b”’s as constants while learning better values for the “a”’s. This can be iterated till the system cannot improve further.



To calculate the value a word has, first add all the b's of the words in the sentence it is in....then multiply this value with its personal "a" value

HATE SPEECH DETECTION NETWORK

Fig 1.1

Another way to describe the network and its associated procedures is that we have an Artificial Neural Network, where we have mapped words to individual neurons instead of having them as variables to be fed into a network. The process of changing training cases will be equivalent to a Dropout procedure where we replace some (or all) of the words/neurons in the previous training case with new ones. Each neuron/word then takes in as input, all the b weights of the other neurons, and weights them all with its personal a weight. To learn this network uses the backpropagation algorithm after calculating an error from the output of an output neuron that will be a traditional neuron. This network then has a unique topology and functions with no inputs. We will use coordinate gradient decent to learn where we alternate between training the a weights of the words and the b weights. The Idealistic Neural Network, is an extremely shallow network that can represent non-linearity in a linear outfit. All of the possible equations a word may have (as derived from the powerset of all English words) are lines or hyperplanes, but the number of equations (here representing the words variables) is still orders of magnitudes above the cardinality of the mapping of the powerset of all English words onto itself. That means the set of possible solutions to any problem we would like to model is infinite and since altogether the NLP task has been thus modelled as a convex optimization problem we may be guaranteed a solution. Since the network has of the order of one million weights at most and is shallow but fully expressive this is a very efficient and easy to train model that we believe will be influential. Other uses could come from treating the three elements in a color model as neurons in a way similar to the way we used individual words. A pixel will then be able to learn its full context in an image (and performing a calculation based off of it for activation) if it receives trained b values from all other regions of the image. To complete this we could place an additional input into each neuron of a constant ranging from 1 upwards into each pixel depending on its position on the image from left to right.