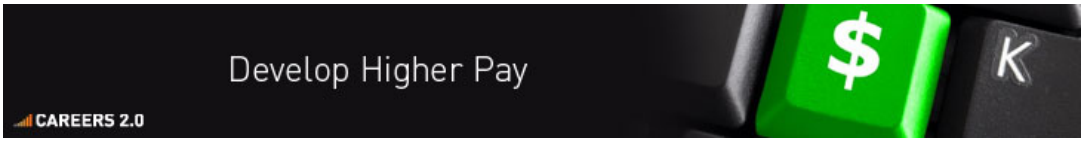
processing strings of text for neural network input



I understand that ANN input must be normalized, standardized, etc. Leaving the peculiarities and models of various ANN's aside, how can I preprocess UTF-8 encoded text within the range of {0,1} or alternatively between the range {-1,1} before it is given as input to neural networks? I have been searching for this on google but can't find any information (I may be using the wrong term).

1. Does that make sense?
2. Isn't that how text is preprocessed for neural networks?
3. Are there any alternatives?

EDIT 20-November-2013:

I have long accepted as correct the answer of Pete. However, I have serious doubts, mostly due to recent research I've been doing on Symbolic knowledge and ANN's.

Dario Floreano and Claudio Mattiussi in their book explain that such processing is indeed possible, by using **distributed encoding**.

Indeed if you try a google scholar search, there exists a plethora of neuroscience articles and papers on how distrubuted encoding is hypothesized to be used by brains in order to encode Symbolic Knowledge.

Teuvo Kohonen, in his paper "*Self Organizing Maps*" explains:

> One might think that applying the neural adaptation laws to a symbol set (regarded as a set of vectorial variables) might create a topographic map that displays the "logical distances" between the symbols. However, there occurs a problem which lies in the different nature of symbols as compared with continuous data. For the latter, similarity always shows up in a natural way, as the metric differences between their continuous encodings. This is no longer true for discrete, symbolic items, such as words, for which no metric has been defined. It is in the very nature of a symbol that its meaning is dissociated from its encoding.

However, Kohonen did manage to deal with Symbolic Information in SOMs!

Furthermore, **Prof Dr Alfred Ultsch** in his paper "*The Integration of Neural Networks with Symbolic Knowledge Processing*" deals exactly with how to process Symbolic Knowledge (such as text) in ANN's. Ultsch offers the following methodologies for processing Symbolic Knowledge: Neural Approximative Reasoning, Neural Unification, Introspection and Integrated Knowledge Acquisition. Albeit little information can be found on those in google scholar or anywhere else for that matter.

Pete in his answer is right about semantics. Semantics in ANN's are usually disconnected. However, following reference, provides insight how researchers have used RBMs, trained to recognize similarity in semantics of different word inputs, thus it shouldn't be impossible to have semantics, but would require a layered approach, or a secondary ANN if semantics are required.

Natural Language Processing With Subsymbolic Neural Networks, Risto Miikkulainen, 1997 Training Restricted Boltzmann Machines on Word Observations, G.E.Dahl, Ryan.P.Adams, H.Rarochelle, 2012

preprocessor | neural-network | textinput | normalize | standardized

edited Nov 20 '13 at 19:44

asked Feb 9 '13 at 0:16

Alex
1,940 2 19 46

1 Whether or not this makes sense depends more on what you're trying to achieve with your ANN. Is your text fixed length? That is, will the input always be the same length string? If not, then this is probably not what you want to be doing. Can you be more descriptive about what you're trying to achieve iwth your ANN in

general? What's the problem you're trying to solve. – Pete Feb 19 '13 at 16:07

@Pete I am trying to parse utf-8 strings into a vector of numbers prior to sending them into the neural network. I do not want feature extraction or compression of any sort, but rather a bi-directional mapping of strings into floats. The reason for this is part of my research on imitation learning and deep belief networks. I cannot get into much detail without writing many pages. My current problem is that I cannot find anywhere any sort of information on how to safely use strings of text (not-fixed length but with a maximum length) as input for ANN. – Alex Feb 19 '13 at 16:17

1 I guess what I'm trying to figure out is, what information about the words is it that you want? Is it their meaning? Is it that you have say 20 words and their particular meaning is unimportant, just which word is associated with the input important? Do you get what I'm asking? Are there a fixed number of words that might be part of your input? I don't think you'll be able to do a real "bidirectional mapping" like you mean, unless the strings are variations of degrees that can be ordered in such a way that the "nearness" of the float value associates with the "nearness" of the words. – Pete Feb 19 '13 at 16:24

@Pete Neither, the words must be given as input without any change. The strings are in essence the input, and must be associated with a specific output. What I am looking for is a way to transform string(s) into an equivalent numerical value that may be processed by the ANN. Just as you map pixels into a representation before giving the vectors as input, same thing. When I say bidirectional, I mean that once the conversion from a string of utf-8 characters to a vector of floats takes place, the reverse should be possible. I am using for UTF-8 library ICU (icu::UnicodeString). – Alex Feb 19 '13 at 16:36

@Pete so far, my thoughts have been to take the decimal code for each UTF-8 Character, and normalize it within -1.0 & 1.0. Since UTF-8 can map 1,111,998 possible combinations, I was intending to get the decimal code for each character found in the string, normalize it, and thus convert it into a vector of floats. Does that make sense ? – Alex Feb 19 '13 at 16:38

add comment

4 Answers

I'll go ahead and summarize our discussion as the answer here.

Your goal is to be able to incorporate text into your neural network. We have established that traditional ANNs are not really suitable for analyzing text. The underlying explanation for why this is so is based around the idea that ANNs operate on inputs that are generally a continuous range of values and the nearness of two values for an input means some sort of nearness in their meaning. Words do not have this idea of nearness and so, there's no real numerical encoding for words that can make sense as input to an ANN.

On the other hand, a solution that might work is to use a more traditional semantic analysis which could, perhaps produce sentiment ranges for a list of topics and then those topics and their sentiment values could possibly be used as input for an ANN.

answered Feb 21 '13 at 16:57

Pete
3,293 13 34

thank you very much for all your help. The only reason I'm not accepting it as an answer yet is because I am hoping for more answers & options. – Alex Feb 21 '13 at 16:59

Well, thanks for the bounty. Hope you find a good overall solution. – Pete Feb 21 '13 at 17:00

add comment

In response to your comments, no, your proposed scheme doesn't quite make sense. An artificial neuron output by its nature represents a continuous or at least a binary value. It does not makes sense to map between a huge discrete enumeration (like UTF-8 characters) and the continuous range represented by a floating point value. The ANN will necessarily act like 0.1243573 is an extremely good approximation to 0.1243577 when those numbers could easily be mapped to the newline character and the character "a", for example, which would *not* be good approximations for each other *at all*.

Quite frankly, there *is no* reasonable representation for "general unicode string" as inputs to an ANN. A reasonable representation depends on the specifics of what you're doing. It depends on your answers to the following questions:

- Are you expecting words to show up in the input strings as opposed to blocks of characters? What words are you expecting to show up in the strings?
- What is the length distribution of the input strings?
- What is the expected entropy of the input strings?
- Is there any domain specific knowledge you have about what you expect the strings to look like?

and most importantly

- What are you trying to *do* with the ANN. This is ***not*** something you can ignore.

Its possible you might have a setup for which there is *no* translation that will actually allow you to *do* what you want with the neural network. Until you answer those questions (you skirt around them in your comments above), it's impossible to give a good answer.

I can give an *example answer*, that would work if you happened to give certain answers to the above questions. For example, if you are reading in strings with arbitrary length but composed of a small vocabulary of words separated by spaces, then I would suggest a translation scheme where you make N inputs, one for each word in the vocabulary, and use a recurrent neural network to feed in the words one at a time by setting the corresponding input to 1 and all the others to 0.

answered Feb 20 '13 at 10:07

Jeremy Salwen
2,414 13 34

1 Thank you. I was trying to figure out how best to explain that. You did an excellent job! – Pete Feb 20 '13 at 14:19

1 @Alex, I think maybe you're not completely understanding Jeremy's explanation. In general, Text is not a useful input for a standard neural network. Again, I'll ask this: What information does the text contain that you want? What is the nature of the information contained in the text that you're trying to capture. The meaning of the words? The letter combinations in the words? This is the point that really needs to be answered before anyone can give you any sort of real answer on this. – Pete Feb 20 '13 at 18:21

2 Okay, then if the meaning of the words is what matters, there no real way to encode it in a standard neural net. Let's say you encode the input to scaled based on ASCII. So each input is a letter. Then, according to your network, DOG and EOG are very similar because, numerically, they'll be very close. But in English, DOG is a word and EOG is just a random combination of letters. There's no nearness. Inputs and outputs are continuous and have a concept of nearness. There is no way to translate text into this sort of framework. – Pete Feb 20 '13 at 18:57

2 Well, again, it's about the meaning of the inputs and outputs. If you encode the words as some kind of value, do the words themselves have property of nearness? That is, if you get an output value that maps to some midpoint between two words (because, remember, your outputs are continuous in the range, not at some fixed intervals), will it still have meaning? What's the meaning of a value half-way between "Good" and "Yellow"? If the meanings in the words can't have some idea of nearness, then that won't work. What you can do is have boolean inputs that represent the existence of a word. – Pete Feb 20 '13 at 21:20

2 @Alex, you might want to investigate Semantic Analysis algorithms. I don't know much about them, but my guess is you could find an algorithm appropriate for your particular needs. Sometimes something like that cna be used to feed an input into the neural network. That is, it might be able to perform an analysis that can give you values which then make sense as input into a neural network model. For example, you might have a topic and a sentiment about that topic. Then a network input could be associated with that topic and the value could be the sentiment, normalized. – Pete Feb 21 '13 at 15:45

show **10** more comments

I think it would be fascinating to feed in text (encoded at the character level) to a deep belief network, to see what properties of the language it can discover.

There has been a lot of work done recently on Neural Network Language modeling (mainly at the word level, but also at the character level)

See these links for more info

http://www.stanford.edu/group/pdplab/pdphandbook/handbookch8.html
http://code.google.com/p/word2vec/

The word vectors are encoded by training on a large corpus of wikipedia articles etc.. and have been able to acquire semantic and syntactic features, which allows a "distance" to be defined between them"

"It was recently shown that the word vectors capture many linguistic regularities, for example vector operations vector('king') - vector('man') + vector('woman') is close to vector('queen')"

Also see this great research paper by Ilya Sutskever on generating random characters, which exhibit the features of the english language after being trained on wikipedia. Amazing stuff!

http://www.cs.toronto.edu/~ilya/pubs/2011/LANG-RNN.pdf http://www.cs.toronto.edu/~ilya/rnn.html (Online text generation text demo - very cool!)

edited Aug 13 '13 at 17:20

answered Aug 13 '13 at 17:13

swami
116 1 9

Hi! Thanks for the input! I've been looking at Self Organizing Maps, as Kohonen in his original paper addressed

the issue of Symbolic Information being processed by ANNs. The first link you've provided seems to use Recurrent Neural Networks, so I can't help but think that maybe a Recurrent Boltzmann Machine may be able to deal with textual input. Thank you for the rest of the links, especially the second one, as I can see my self using it soon. Regards, Alex. – Alex Aug 16 '13 at 14:30

add comment

It is not exactly clear what you are trying to do, but I guess that it seems to be in some sense related to what people call "Natural Language". There are lots of references about this... I am not an expert, but I know for example that there are some interesting references by O'Reilly.

From the NN perspective there are lots of different NN models. I think you are referring to the most popular one known as Multilayer perceptron with a kind of backpropagation algorithm, but there are lots of models of associative memory that may be more suitable for your case. A very good reference about this is the Simon Haykin book.

However, if I tried to do something like this, I would start trying to understand how the frequency of letters, syllables and words arise together in English language (?).

I hope that I helped. As I told before, I am not an expert in the field.

answered Feb 20 '13 at 20:49

DanielTheRocketMan
1,023 3 19

1 Thank you, yes it does have to do with Natural Language processing since input is received in natural language, although the actual point is extracting information (or knowledge) from the natural language and associating it with a solution. Why do you mention frequency of letters and syllables (you are not the first person to tell me that)? – Alex Feb 20 '13 at 20:59

1 I am not sure what you are trying to do but having sad that... In every language, letters arise in different frequencies. See for instance en.wikipedia.org/wiki/Letter_frequency... If there is a missing letter in your word and you do not have any other information, you could simply use the frequency of the letters in the English language. If you have the syllable, you have more information. Syllables also arise in different frequencies... If you are talking about complete clauses, you know that different words have different functions and again arise in different frequencies... – DanielTheRocketMan Feb 20 '13 at 22:40

add comment

Not the answer you're looking for? Browse other questions tagged preprocessor | neural-network | textinput | normalize | standardized **or ask your own question.**