Department of Computing

Imperial College of Science, Technology and Medicine

Introduction Lecture for NLP and some ML baselines

THe introductory lecture for Natural Language Processing and some core concepts for Natural Language Processing with supplementary material regarding basic ML concepts required for this course

Author: Anton Zhitomirsky

Contents

1 Dealing with natural language

1.1 Complexities

- Ambiguity at word level

  "Can you bring me the file"

- Syntactic ambiguity (prepositional phrase attachment ambiguity)

  "I saw the boy with a telescope"

- Semantic ambiguity

  "I haven't slept for 10 days" "The rabbit is ready for lunch"

- Referential ambiguity

  "We gave the monkeys bananas because they were more than ready to eat."

- Non-literal meaning

  "Call me a cab, it's raining cats and dogs"

1.2 The role of Deep Learning

- The creation and maintenance of linguistic rules often is infeasible or impractical.
- We'd much rather learn functions from data instead of creating rules based on intuition.
- deep learning is very flexible, learnable framework fro representing information from many different modalities.

1.3 Composites of Language

1.3.1 Lexicon - morphological analysis

Definition: Words: segmentation, normalisation, morphology

1. Word Segmentation (tokenization, decompounding):

   "For example, most of what we are going to do with language relies on first separating out or tokenizing words from running text, the task of tokenization. English words are often separated from each other tokenization by whitespace, but whitespace is not always sufficient. "New York" and "rock 'n' roll" are sometimes treated as large words despite the fact that they contain spaces, while sometimes we'll need to separate "I'm" into the two words "I and am"" [2]

2. Word normalization (capitalization, acronyms, spelling variants)
3. Lemmatisation (reduce to base form = valid word)

   "Another part of text normalization is lemmatization, the task of determining lemmatization that two words have the same root, despite their surface differences. For example, the words sang, sung, and sings are forms of the verb sing. The word sing is the common lemma of these words, and a lemmatizer maps from all of these to sing" [2]

4. Stemming (reduce to root = not always valid word)

   "Stemming refers to a simpler version of lemmatization in which we mainly stemming just strip suffixes from the end of the word" [2].

5. Byte-pair encoding (BPE) and wordpieces

   "Instead of defining tokens as words (whether delimited by spaces or more complex algorithms), or as characters (as in Chinese), we can use our data to automatically tell us what the tokens should be" [2].

function BYTE-PAIR ENCODING(strings C, number of merges k) **returns** vocab V

$V \leftarrow$ all unique characters in C # initial set of tokens is characters
for $i = 1$ **to** k **do** # merge tokens k times
  $t_L, t_R \leftarrow$ Most frequent pair of adjacent tokens in C
  $t_{NEW} \leftarrow t_L + t_R$ # make new token by concatenating
  $V \leftarrow V + t_{NEW}$ # update the vocabulary
  Replace each occurrence of t_L, t_R in C with t_{NEW} # and update the corpus
return V

Figure 2.13 The token learner part of the BPE algorithm for taking a corpus broken up into individual characters or bytes, and learning a vocabulary by iteratively merging tokens. Figure adapted from Bostrom and Durrett (2020).

   "The BPE token learner begins BPE with a vocabulary that is just the set of all individual characters. It then examines the training corpus, chooses the two symbols that are most frequently adjacent (say 'A', 'B'), adds a new merged symbol 'AB' to the vocabulary, and replaces every adjacent 'A' 'B' in the corpus with the new 'AB'. It continues to count and merge, creating new longer and longer character strings, until k merges have been done creating k novel tokens; k is thus a parameter of the algorithm. The resulting vocabulary consists of the original set of characters plus k new symbols" [2].

6. Part-of-speech tagging (Recognize category of word): verb, noun, adverb, adjective, determiner, preposition...

   It is the process of assigning a part-of-speech to each word in a text. This is a disambiguation task; words are ambiguous - have more tha one possible part-of-speech - and the goal is to find the correct tag for the sitauation. E.g. 'book' can be a verb ('book that flight') or a noun ('hand me that book'). The POS-tagging resolves these ambiguities by choosing the proper tag for the context [2].

7. Morphological analysis (recognize/generate word variants):

   "Morphology is the study of the way words are built up from smaller meaning-bearing units called morphemes. Two broad classes of morphemes can be distinguished: **stems** — the central morpheme of the word, supplying the main meaning — and **affixes** — adding "additional" meanings of various kinds. So, for example, the word fox consists of one morpheme (the morpheme fox) and the word cats consists of two: the morpheme cat and the morpheme '-s'" [2]

1.3.2 Syntax

Syntax comes from the Greek s'yntaxis, meaning "setting out together or arrangement", and refers to the way words are arranged together [2].

We can form a parse tree based on some syntax rules which makes up grammar. This was typically used in applications before deep learning; there are sentences that make sense but don't follow grammar.

	Phrase Structure Rule	Example
$S \rightarrow NP \quad VP$	Sentence → Noun-phrase Verb-phrase	I prefer a morning flight
$NP \rightarrow Det\ N$	Noun-phrase → Determiner Noun	prefer a morning flight
$VP \rightarrow VNP$	Verb-phrase → Verb Noun-phrase	leave Boston in the morning
$VP \rightarrow V$	Verb-phrase → Verb	
$VP \rightarrow VPP$	Verb-phrase → Verb Propositional-phrase	leaving on Thursday
$PP \rightarrow PNP$	Preposition-phrase → Preposition Noun-phrase	from Los Angeles

1.3.3 Semantics

Definition: Meaning of words and sentences

"We also introduce word sense disambiguation (WSD), the task of determining which sense of a word is being used in a particular context [...] A sense (or word sense) is a discrete representation of one aspect of the meaning of a word." [2].

Compositional meaning understands who did what to whom, when, where, how and why. It composes the meaning of the setnece, based on the meaning of the words and the structure of the sentence. Here, the dog chased the man = The man was chased by the dog, but The dog bit the man $\neq$ The man bit the dog.

"Semantic role labeling (sometimes shortened as SRL) is the task of automatically finding the semantic roles of each argument of each predicate in a sentence" [2].

1.3.4 Discourse

Definition: Meaning of a text (relationship between sentences)

"language does not normally consist of isolated, unrelated sentences, but instead of collocated, structured, coherent groups of sentences. We refer to such a coherent structured group of sentences as a discourse, and we use the word coherence to refer to the relationship between sentences that makes real discourses different than just random assemblages of sentences" [2].

"Coreference resolution is the task of determining whether two mentions corefer, by which we mean they refer to the same entity in the discourse model (the same discourse entity)" [2].

1.3.5 Pragmatics

Definition: Intentions, commands; what is the intent of the text, how to react to it?

2 How to represent language to an algorithm

2.1 One-hot-encoding

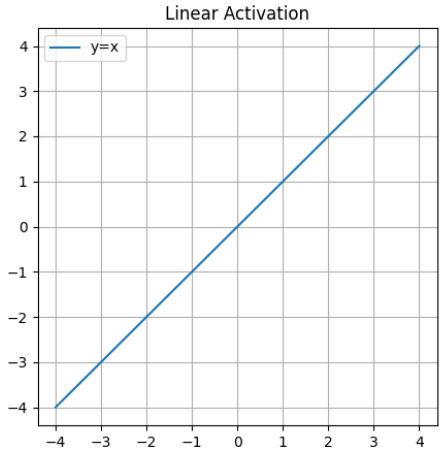
- Corpus: A collection of documents, i.e. our entire dataset
- Document: one item of our corpus (e.g. a sequence)
- token: the atomic unit of a sequence (e.g. a word)
- vocabulary: the unique tokens across our entire corpus

3 ML Refresher

3.1 Linear activation function

Definition 3.1 (Linear Activation)**.**

$$f(x) = c \cdot x$$



- This produces a constant gradient, meaning that during backpropagation, the updates applied to weights are constant and independent of the change in input, denoted by Δx.
- If each layer in a multi-layered network employs a linear activation function, the output of one layer becomes the input to the next, perpetuating linearity throughout the network; no matter how many layers you have, the entire network behaves like a single-layer linear model. This means you could replace all N linear layers with just a single linear layer and achieve the same output. It renders the "depth" of the network irrelevant.
- Useful for regression

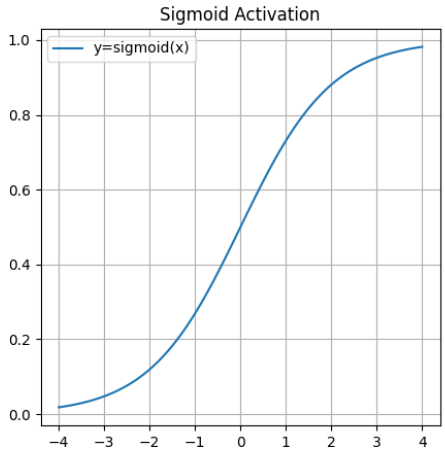
Therefore, while linear activation functions may have some use-cases, they aren't typically chosen for complex machine learning tasks that require the network to capture more complex, non-linear relationships in the data.

3.2 Non-linear activation functions

3.2.1 Sigmoid

Definition 3.2 (Sigmoid Activation)**.**

$$f(x) = \frac{1}{1 + e^{-x}}$$

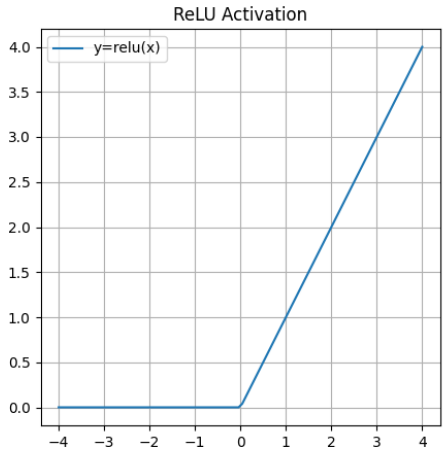


- This function is non-linear, allowing us to stack layers in a neural network, thereby facilitating the learning of more complex representations.
- gives a more analog or continuous output w.r.t binary step function.
- Smooth gradient which is crucial for gradient descent algorithms. One notable characteristic is that between the X values of -2 and 2, the curve is especially steep. This implies that small changes in the input within this region result in significant shifts in output, facilitating rapid learning during the training phase.
- Towards the tails of the function, the curve flattens out, and the output values become less sensitive to changes in input. This results in a vanishing gradient problem, where gradients become too small for the network to learn effectively, leading to slow or stalled training.
- Good for classifiers (binary and multi-label classification)

3.2.2 ReLU

Definition 3.3 (ReLU Activation)**.**

$$f(x) = max(0, x)$$



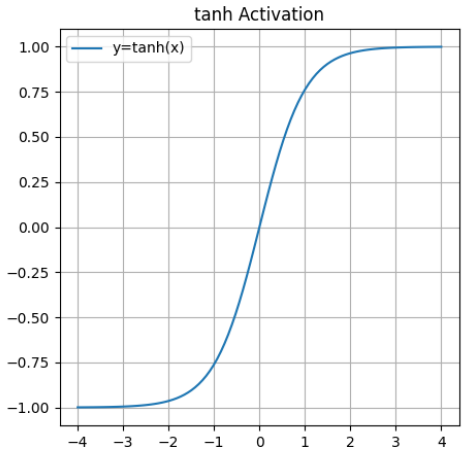
- piecewise linear that outputs the input directly if it is positive, otherwise, it outputs zero

- ReLU is inherently non-linear when considered as a whole, particularly due to the sharp corner at the origin.
- combinations of ReLU functions are also non-linear, enabling us to stack layers in neural networks effectively. Because it is a universal approximator.
- unbounded as $[0, \infty)$
- unboundedness can cause explosions of activations if not managed properly
- tends to produce sparse activations
- In a neural network with many neurons, using activation functions like sigmoid or tanh would cause almost all neurons to activate to some degree, leading to dense activations. ReLU, on the other hand, will often output zero, effectively ignoring some neurons, which can make the network more computationally efficient.
- Dying ReLU problem caused by often outputting zero (If a neuron's output is always zero (perhaps due to poor initialization), the gradient for that neuron will also be zero.) As a result, during backpropagation, the weights of that neuron remain unchanged, effectively "killing" the neuron. This can result in a portion of the neural network becoming inactive, thereby limiting its capacity to model complex functions.

3.2.3 Tanh

Definition 3.4 (Tanh Activation)**.**

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

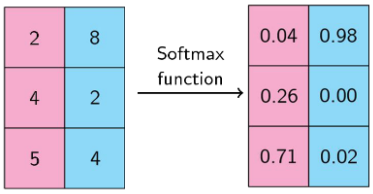


- Scaled version of sigmoid but from -1 to 1 instead of 0 to 1
- non-linear, we can stack it
- since $(-1, 1)$ less concern about activations becoming too large and dominating the learning process
- One key benefit of tanh over sigmoid is that its gradient is stronger; that is, the derivatives are steeper. This can make it a better choice for certain problems where faster convergence is desired.

- its outputs are zero-centered, meaning the average output is close to zero. This is beneficial for the learning process of subsequent layers, as it tends to speed up convergence by allowing for a balanced distribution of outputs and gradients.

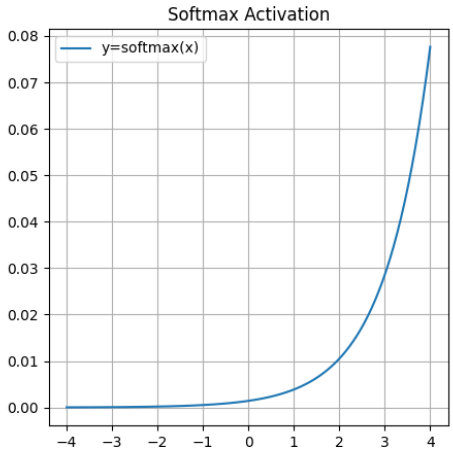
However, like the sigmoid function, tanh also suffers from the vanishing gradient problem when you stack many layers, which can slow down learning.Careful normalization of the inputs is also essential when using tanh to ensure effective learning.

3.2.4 Softmax



Definition 3.5 (Softmax Ativation)**.**

$$softmax(z_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$



- applies softmax to an n-dimensional input tensor rescaling them so that the elements of the n-dimensional output tensor lie in the range $[0, 1]$ and sum to 1
- when dealing with image classification, the Softmax activation function is commonly employed to transform the network's logits into probabilities.
- useful for multi-label classification (predicting one class out of many)

3.3 Loss Functions

3.3.1 Mean suqared error

Definition 3.6 (L2 Norm, mean squared error)**.**

$$\ell(x, y) = \boldsymbol{\mathcal{L}} = \{l_1, \dots, l_N\}^T, \quad l_n = (x_n - y_n)^2$$

With further reduction to a single value can be either $mean(\mathcal{L})$ or $sum(\mathcal{L})$

- Measured discrepancy between the predicted output and the actual ground truth
- in a mini-batch, perform squared difference calculation for each sample within the mini-batch, the individual squared errors can then be combined into a list.
- Useful for regression

3.3.2 Binary cross-entropy

Definition 3.7 (Binary Cross Entropy)**.**

$$\ell(x, y) = \mathcal{L} = \{l_1, \dots, l_N\}^T,$$
$$l_n = -w_n[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n)]$$

- x_n represents the predicted probability for the positive class
- y_n is the ground truth label (0 or 1) for that observation
- w_n is a weights associated with that sample

With further reduction to a single value can be either $mean(\mathcal{L})$ or $sum(\mathcal{L})$

- negative weighted sum of two entropy terms for each observation in the batch
- the input probabilities, x_n, must lie in the [0, 1] range to ensure the loss's validity
- Suitable for binary classification tasks where each observation belongs to one of two classes.
- Useful for binary classification
- Useful for multi-label classification (predicting many classes)

3.3.3 Categorical cross-entropy

useful for multi-label classification (predicting one class out of many)

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_c^{(i)} \log(\hat{y}_c^{(i)}) \tag{1}$$

3.4 Regularization

any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error. (See Chapter 7 of [1])