Parsing Algorithm

An algorithm for encoding data using a dictionary of parts

Overview

Assume there's a dictionary of atomic parts, for example $\triangle = \{spr, spri, ing, g\}$.

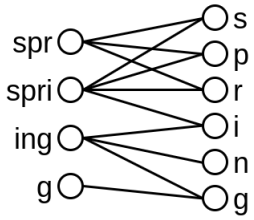
And, let's say we're analyzing data, such as $x =$ "spring".

The activation matrix maps dictionary elements to parts of the data, as shown below:

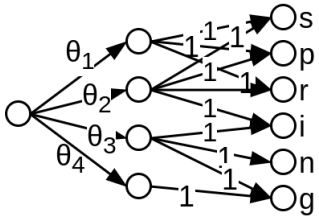
$$M = \begin{array}{c} \\ spr \\ spri \\ ing \\ g \end{array} \begin{array}{c} \begin{array}{cccccc} s & p & r & i & n & g \end{array} \\ \begin{pmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 \\ \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} \end{pmatrix} \end{array} \tag{1}$$

$$M \in \mathbb{R}^{d \times |x|}$$

Graphically, the matrix represents the associations in a bipartite graph:



Consider a network flow, where the weights $\theta_1, ..., \theta_d$ select the dictionary elements.



$$\theta = (\theta_1, \theta_2, ..., \theta_d) \in \mathbb{R}^{1 \times d}$$

Learning

The output of the network flow is

$$y_{1\times|x|} = \theta M \tag{2}$$

An optimal encoding of the data (using the dictionary) should activate every part of the data, meaning the outputs of the network should all be at least 1.

$$y \geq 1 \tag{3}$$

However, encodings are mutually exclusive, so an activation should not be greater than 1.

$$y \leq 1 \tag{4}$$

θ is approximated by minimizing the loss function using gradient descent:

$$Loss(\theta) = \|y - 1_{1\times d}\|_2 \tag{5}$$

Experiments

I evaluated the network approach on 1D text examples designed by Si et al. (2013).

x_1 = "xjcdspringkgfkis nowsgrwaejsbcominghdzxv"

↳ Greedy parse: ['spri', ' ', 'is n', ' ', 'comi']

↳ ParseNet: ['spr', 'ing', 'is ', 'now', 'com', 'ing']

x_2 = "hvwinterjvvshbrjkwas nowudlwgcolderiutjrqkvjg"

↳ Greedy parse: ['wint', ' ', 'was ', 'now', ' ', 'cold']

↳ ParseNet: ['win', 'ter', 'was ', 'now', 'col', 'der']

x_3 = "hvwinterjvvshbrjkwas nowudlwgcolderiutjrqkvjg"

↳ Greedy parse: ['hams', 'ter', ' ', 'is n', ' ', 'jump', 'ing']

↳ ParseNet: ['hams', 'ter', 'is ', 'now', 'jum', 'ing']