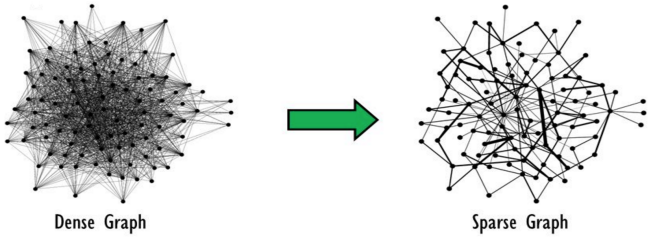
Sparse Coding

Arya Grayeli

April 2020

1 Introduction

Sparse coding is the study of algorithms which aim to learn a useful sparse representation of any given data. This sparse representation is a lot smaller than the data and only targets relevant information (signals), excluding irrelevant information (noise). It works by encoding each datum as sparse code. This does not require any output data, making it a form of unsupervised learning. In addition, sparse coding resembles how the neurons in our brain work. We have billions of neurons but for a specific task only certain neurons fire. Hence, the goal of sparse coding is to find the relevant "neurons" for a task and to strongly activate that small subset, leaving the rest of the "neurons" idle.



2 Uses

Since sparse coding is very good at finding the intrinsic information, it is used in processes that require the noise to be filtered out. These include image processing, audio processing, NLP and stock chart analysis. Aside from denoising, sparse coding is useful since it compresses the data, making understanding, transmitting and storing data much easier.

3 Sparse Modeling

3.1 Basics

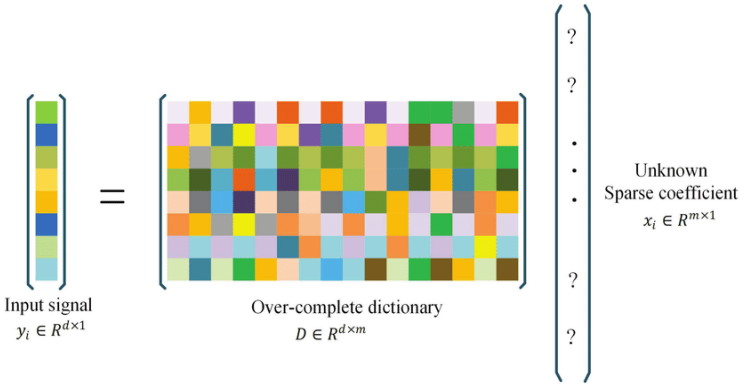
The aim of sparse modeling is to represent the input data x as,

$$x = \sum_{i=1}^{m} a_i \Phi_i$$

where Φ_i are basis vectors and a_i are coefficients. Another way to represent the above equation is as,

$$x = D\alpha$$

where x is the d by 1 input vector, D is the dictionary matrix (d by m) and α is a m by 1 vector that is the sparse representation. A visual example of this is below:



Since we are given x, our goal is to find the appropriate D and α such that they can approximately represent x while also being sparse. This leads us to the formal definition.

3.2 Formal Definition

$$x \approx D\alpha \text{ for some } \alpha \in \mathbb{R}^m \text{ with } \|\alpha\|_0 \ll m$$

The first part shows how D and α should model x, and the second part shows that α needs to be sparse. In addition, D should be over-complete, meaning d $<$ m. So formally, the goal of sparse modeling is,

1. Find an over-complete dictionary D
2. Find a sparse representation α

such that the property above is satisfied.

3.3 Optimization

Using the information stated above, this problem can be represented as finding the optimal solution to the formula below,

$$\underset{\mathrm{D} \in C,\, \alpha \in \mathbb{R}^m}{\operatorname{argmin}} \sum_{i=1}^{d} \|x_i - D_i\alpha\|_2^2 + \lambda \sum_{i=1}^{m} \|\alpha_i\|_0 \text{ where}$$
$$C \equiv \{D \in \mathbb{R}^{d \times m} : \|D_i\|_2 \leq 1 \ \forall i = 1, ..., d\}, \lambda > 0$$

In words, the formula above is stating to find the minimum sum of the error and sparsity. C is simply a constraint on D such that the values in the dictionary (called atoms) are not too large, so then the values inside α are low. In addition, λ is the sparsity coefficient.

The problem with the above formula is that it is NP-hard due to the L_0 norm in the sparsity section ($\lambda \sum_{i=1}^{m} \|\alpha_i\|_0$). NP-hard stands for non-deterministic polynomial time hardness and means that the problem cannot be solved in polynomial time, making it very slow and expensive. So that's why the formula used in application replaces the $\lambda \sum_{i=1}^{m} \|\alpha_i\|_0$ with $\lambda \sum_{i=1}^{m} \|\alpha_i\|_1$ since the L_1 norm solves the NP-hard problem while maintaining the sparsity aspect in most cases. Another commonly used replacement is $\lambda \sum_{i=1}^{m} \log(1 + {\alpha_i}^2)$. This change makes it a convex optimization problem with D and α, however only when one is fixed, meaning we cannot solve for both at the same time (jointly). This leads us to many algorithms to solve for D and α separately.

4 Algorithms

4.1 Matching Pursuit

Matching pursuit finds the optimal α given x and D.

Input: Signal x, dictionary D with normalized columns g_i
Output: List of coefficients $(a_i)_{i=1}^m$ and indices for corresponding atoms $(\gamma_i)_{i=1}^m$

Initialization:
$R_1 \leftarrow x$
$i \leftarrow 1$

Repeat:
Find $g_{\gamma_i} \in D$ with maximum inner product $|\langle R_i, g_{\gamma_i} \rangle|$;
$a_i \leftarrow \langle R_i, g_{\gamma_i} \rangle$;
$R_{i+1} \leftarrow R_i - a_i g_{\gamma_i}$;
$i \leftarrow i + 1$;
Until stop condition (ex. $\|R_i\| <$ Threshold)
return

This algorithm works since R_i converges to 0. In addition, we need to use the a and γ values to create the α vector. The downside of matching pursuit is its computational complexity.

4.2 Method of Optimal Directions (MOD)

MOD addresses both updating α and D in an alternating fashion. It does this by initializing D and α and then repeating these two steps:

1. Use matching pursuit to find α
2. Update D using $D = x\alpha^+$ where α^+ is the Moore-Penrose pseudo-inverse of α and then re-normalizing D

This process goes on until it converges (or reaches a small residue). A problem with MOD is the high complexity of computing the Moore-Penrose pseudo-inverse.

4.3 Other Algorithms

There are many more algorithms for finding the appropriate D and α. They include:

- K-SVD
- Stochastic Gradient Descent
- Lagrange Dual Method
- LASSO

5 Conclusion

Sparse coding is a method that can help limit the number of values your model has to deal with, only finding the intrinsic nodes such that the output is a lot smaller. Due to these capabilities, sparse coding is useful in denoising and condensing data. It does this by representing the input data as the product of an over-complete dictionary (D) and a sparse set of coefficients/weights (α). In order to find the optimal D and α, various algorithms are used. However, one disadvantage of sparse coding is the time it takes to optimize D and α since for every new input it needs to go through the same learning process. This makes its run time (even during testing) significantly large. On the other hand, it makes the subsequent calculations a lot faster due to the smaller output. These trade-offs need to be taken into account when deciding between a dense and sparse model.