Loopy Belief Propagation

for image processing

Chistuakov Alexander

MSU, CMC 201 group

Introduction

In modern science and industry challenges associated with the description of random factors influence are often. Such problems appear in the fields of statistical physics, computer vision, coding and decoding messages and models of artificial intelligence.

Usually, the model which describes the stated challenge can be simulated with a special graph (graphical model). In this model vertexes are responsible for local states of variables and edges describe the influence between variables. One of the major problems of most of these challenges is that the solution can not be found except a complete listing of all possible states. As an example, these NP-complete problems are the selection of a variable marginal distribution and finding the most probable state of the system.

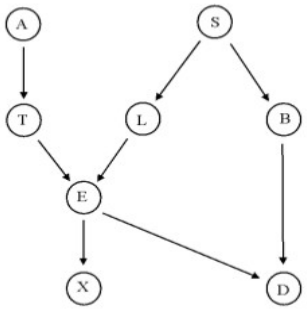
However, in some cases, the solution can be accelerated with algorithms, which does not guarantee to find the right answer, but usually give an answer close to the right. One of these algorithms is considered in this article – loopy belief-propagation algorithm (LBP).

We will consider the principle of belief-propagation algorithm (BP), explore the possibility of it's implementation optimizations and mark the set of problems LBP can be used for.

Main graphical models

Before we start to study BP algorithm itself, let's specify the model of the graph for which the algorithm will be applied. In the course of graphic models the following three types types are most important:

Bayesian Network – directed acyclic graphs whose nodes represent random variables, and edges represent conditional dependencies.



To calculate a probability of the whole system's statement $\{x\}$ you should multiply the probabilities of states for each vertex according to their's conditional probabilities.

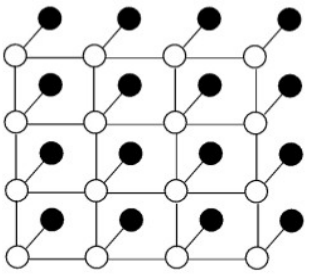
$p(x_1, x_2, \ldots, x_N) = \prod_{i=1}^{N} p(x_i | Par(x_i))$, where $Par(x_i)$ – parents of the vertex i.

For the network showed above we can write it as

$$p(\{x\}) = p(x_A) \cdot p(x_S) \cdot p(x_T | x_A) \cdot p(x_L | x_S) \cdot p(x_B | x_S) \cdot p(x_E | x_L, x_T) \cdot p(x_X | x_E) \cdot p(x_D | x_E, x_B)$$

As an example, this model is good for describing the system of influence between diseases and symptoms.

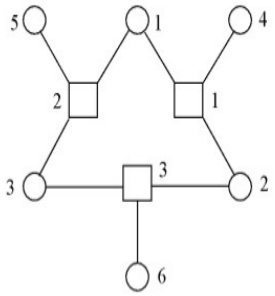
Pairwise Markov Random Field – this model is rather similar to Bayesian network. However in this model all edges are undirected and each vertex has it's own prior distribution.



We can image it with an analogy of Ising model. Our graph will be the regular grid (squared for example), each vertex – is an abstract polarized particle, which can affect with positive or negative charge. Then we can introduce functions $\psi_{ij}(x_i, x_j)$ named interaction for each pair of neighbor vertexes and the function $\phi_i(x_i)$ named the field potential for each vertex.

So the joint probability of statement $\{x\}$ can be calculated as $p(\{x\})=\frac{1}{Z}\prod_{(i,j)}\psi_{ij}(x_i,x_j)\prod_i\phi_i(x_i,y_i)$, where $\frac{1}{Z}$ – is a normalizing constant .

Factor graph – a bipartite graph representing the factorization of a function. It contains vertexes of two kinds: variables, which have theirs prior distributions $\phi_i(x_i)$, and factors which describes the probability of all it's neigbor statement $\psi_f(|neig(f)|)$



The joint probability of statement $\{x\}$ is $p(\{x\})=\frac{1}{Z}\prod_f\psi_f(|neig(f)|)\prod_i\phi_i(x_i)$, where $\frac{1}{Z}$ – as a before, a normalizing constant.

One of the problems the factor graph could be applied for – a message decoding.

The unification of graphics models

The article [1] contains an algorithm which afford to process equivalent modifications between all the models listed above. It gives us a possibility to transform all the models to the Markov Random Field (MRF) and discuss only them.

This transformation is reasonable by the two reasons:

1. The BP algorithm we are going to discuss is based on sending of special messages between different vertexes over the graph. The Bayesian network will make us to mark the difference between messages from parents to sons and from sons to parents. The Factor graph model will differ messages from factors and from variables. In opposite to these two models, the Markov Random Field has a complete symmetry and make no difference between all the messages.
2. The main cause of this article – is a demonstration of applying of BP algorithm for image processing. The MRF is the most natural model to adapt to this problem.

General description of BP algorithm

Let us set, that we have an undirected graph *G* with functions from MRF model. And we want to find out the marginal distribution of a certain variable. Moreover, we suppose, that each vertex has a finite number of different states. (If not, we can always split the definition area on a finite number of segments and then continue our work with a small error).

According to a standard theory, if we want to calculate the probability of statement x_i for vertex *i*, we must sum the probabilities over all the system statements, contains x_i .

$$p(x_i)=\sum_{x_1}\sum_{x_2}\dots\sum_{x_{i-1}}\sum_{x_{i+1}}\dots\sum_{x_N} p(x_1, x_2, \dots, x_N)$$

$$p(x_i)=\frac{1}{Z}\sum_{\{x\}\ni x_i}\prod_{(u,v)}\psi_{uv}(x_u, x_v)\prod_{u}\phi_u(x_u)$$

The complexity of calculations grows exponential and could not be used for real problems.

To solve it for many cases we can use the BP algorithm. The simplest variant of its implementation is described bellow.

Let us process *K* iterations, and on each of them all the pairs of neighbor vertexes will send some special messages to each other. The message $\mu_{v\to u}$ is a vector the same length as the vertex *u* has states and describes how the vertex *v* "beliefs", that vertex *u* can be stayed in these statements. To calculate the certain coordinate of such message we use formula:

$$\mu_{v\to u}^{new}(x_u)=\sum_{x_v}\psi_{vu}(x_v, x_u)\cdot\phi_v(x_v)\prod_{k\in neig(v)\setminus u}\mu_{k\to v}^{old}(x_v)$$

where $neig(v)$ – all the neighbor of vertex *v*. Initially all the messages could be random.

After all, we can calculate the vertex distribution $p(x_i)$ as

$$p(x_i)=\frac{1}{Z}\cdot\phi_i(x_i)\cdot\prod_{k\in neig(i)}\mu_{k\to i}(x_i)$$

It is not difficult to show, that this algorithm gives the correct distribution for each variable after $diameter(G)\leqslant N$ iterations. For random graphs there is no proof of this algorithm converges, but on practice in many cases it work good. In cyclic graphs BP algorithm called Loopy BP.

Max-possibility statement search

Before we discuss the announced applying of BP algorithm, lets mark another

useful property of the algorithm. If we will only change the *sum*-operator on *max*-operator we will learn to find the most possible statement $\{\tilde{x}\}$ of the whole system

$$p(\{\tilde{x}\}) = \max_{x_1} \max_{x_2} \dots \max_{x_N} p(\{x\})$$

New message updating formula will look like

$$\mu_{v \to u}^{new}(x_u) = \max_{x_v} \psi_{vu}(x_v, x_u) \cdot \phi_v(x_v) \prod_{k \in neig(v) \setminus u} \mu_{k \to v}^{old}(x_v)$$

The formula for calculating of the single-vertex probabilities will left without changing. After that, to find the most possibility statement we just need to mark the statement with the highest "probability" for each vertex.

Image processing

Now, let us talk about the promised image processing by using BP . The first problem we learn to solve is image restoring. For example we have a bitmap image received with some sharps and noises and we want to restore the initial image.

To do it, we can perform this image as a regular graph and associate each vertex with an random-variable possible to stand in *k* different states (brights of the pixel). Then we introduce a penalty for a big variance from initial image and for differences between neighbor colors by using functions from MRF. As an example $\phi_i(x_i) = e^{c_1 \cdot |x_i - x_{i0}|}, \psi_{ij} = e^{c_2 \cdot |x_i - x_j|}$. As experiments show, the LBP algorithm converges very fast on such graphs.

If we want to solve another problem – the projecting of stereo vision on a plane, we can just change the potential function: $\phi_i(x_i) = (e^{c \cdot |x_i - L_i|} + e^{c \cdot |x_i - R_i|})/2$, where L_i and R_i are colors of pixels in two initial left and right images.

Finally, some similar formulas could be used for image segmentation in recognition problems.

Another problems BP can solve

Today the BP algorithm is widely spread in areas of Bayesian prognosis, image and stereo image processing, physic modulation and graph coloring.

Summary this algorithm gives the correct answer on tree-graphs and usually close to correct answers on random graphs.

Main advantages are simplicity in implementation and ease to parallelize.

References

[1] Jonathan S. Yedidia. Understanding Belief Propagation and its Generalizations.

[2] Pedro F. Felzenszwalb . Efficient Belief Propagation for Early Vision .