Scene Text Recognition

Vincent BODIN & Thomas MOREAU

Scene text recognition refers to finding automatic ways of extracting text in pictures of every day life. As computer vision is more and more successful, scene text recognition has naturally become a significant issue nowadays. Unlike OCR which is more or less well understood and implemented, scene text recognition still needs many improvement to be considered as a powerful tool. We implemented paper [1] that explains how to extract words efficiently. A graphical model is introduced inside, with the creation of graph represented possible detected words, and algorithm TRW-S [2] is used to extract the optimal one.

1 Word recognition

This first part of the project has been done as a project for the course *Object recognition and computer vision*. It consists in:

1. Train classifiers: for every possible character in the English language, train a SVM with a RBF kernel (one-versus-all procedure). The database used is ICDAR 2003 [3]. Evaluate on a dictionary the frequency of occurrence of each pair of characters.
2. Recognizing character: performed via sliding window detection. For each possible window, compute the likelihood of each character and keep it if it has a confidence high enough.
3. Building words: build a graph of windows close enough in the x-direction. Attribute an energy to each possible word an energy depending on the lexicon prior.
4. Minimization of CRF energy: minimize this discrete energy using TRW-S algorithm [2].

The main apsect of this project is how to build the graph and minimize its discrete energy.

PICTURES

2 TRW-S Algorithm

2.1 Preliminaries

Introduced by V. Kolmogorov [2], it minimizes a discrete energy on a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$:

$$E(\mathbf{x}|\theta) = \theta_c + \sum_{s \in \mathcal{V}} \theta_s(x_s) + \sum_{(s,t) \in \mathcal{E}} \theta_{st}(x_s, x_t) \tag{1}$$

For a general graph, it is NP-hard. In [4], *belief propagation* was described - which gives the exact optimum for trees, but might get stuck into a loops otherwise. In [5] *max-product tree-reweighed message passing* (TRW) is developed. *Sequential tree-reweighted message passing* (TRW-S), [2] is better because the bound is tighter.

2.2 Belief Propagation, [4]

It is dynamic programming in linear time that is exact for trees. Denote by $\mathcal{X}_s$ the labels that can take node s, and $x_s \in \mathcal{X}_s$.

Step 1: Messages are going from leaves to root:

$$M^{s \to t}(x_t) = \min_{x_s} \left(\theta_s(x_s) + \theta_{s,t}(x_s, x_t) \right), \; x_j \in \mathcal{X}_t \tag{2}$$

Step 2: p10 ppt.

2.3 Algorithm

Given a graph $\mathcal{G}$ we extract a collection of trees $\mathcal{T}$ with a certain distribution $\rho^T, T \in \mathcal{T}$. We define θ^T as a vector θ compatible with T - *i.e.* a which is null outside T. Set:

$$\mathcal{A} = \{\theta \in \mathbb{R}^{d \times |\mathcal{T}|} | \theta^T \text{ for all } T \in \mathcal{T}\} \tag{3}$$

We denote by $\theta' \equiv \theta$ if they give the same energy E for all $\mathbf{x}$. Then the algorithm TRW-S consists in solving:

$$\max_{\theta \in \mathcal{A}, \sum_T \rho^T \theta^T \equiv \bar{\theta}} \Phi_\rho(\theta) = \max_{\theta \in \mathcal{A}, \sum_T \rho^T \theta^T \equiv \bar{\theta}} \sum_T \min_{x \in \mathcal{X}} \left\langle \theta^T, \phi(x) \right\rangle \tag{4}$$

Petit encart publicitaire

Pub pub pub