Advanced Computer Vision

Практический курс

Савельева Юлия Олеговна, 3 семестр, 06.10.2022

Text Recognition

Градиент целевой функции

Реализовать вычисление градиента целевой функции ->

lh[t] = $$\sum_{s=1}^{|\mathbf{l}'|} \frac{\alpha_t(s)\beta_t(s)}{y^t_{\mathbf{l}'_s}}$$

lab[t] = $$\sum_{s \in lab(\mathbf{z},k)} \hat{\alpha}_t(s)\hat{\beta}_t(s)$$

out[i][t] = $$y^t_k$$

Algorithm 2: CTC Loss and **softmax** gradient computation

Data: $\text{out}_{|\hat{A}| \times (\bar{W}_{padded}/4)}$ (result of softmax), where $|\hat{A}|$ (alphabet size);
$T = \bar{W}_{unpadded}/4$;
label (encoded by alphabet);
$bl = 0$ (blank index)
begin
  $L = 2 \times len(label) + 1$
  $out_{unpadded} = zeros(|\hat{A}|, T)$
  for $t := 0$ **to** T **do**
    for $i := 0$ **to** $|\hat{A}|$ **do**
      $out_{unpadded}[i][t] = out_{padded}[i][t]$
  $a = ComputeAlpha(out_{unpadded}, label, bl)$
  $out^{flipped}_{unpadded} = fliplr(out_{unpadded})$
  $label_{reversed} = reverse(label)$
  $b = ComputeAlpha(out^{flipped}_{unpadded}, label_{reversed}, bl)$
  $b = flipud(fliplr(b))$
  $ab = a * b$
  $lab = zeros(|\hat{A}|, T)$
  for $s := 0$ **to** S **do**
    $i = max(0, floor(\frac{s-1}{2}))$
    if s ***mod*** $2 = 0$ **then**
      for $t := 0$ **to** T **do**
        $lab[bl][t] = lab[bl][t] + ab[s][t]$
        $ab[s][t] = \frac{ab[s][t]}{out_{unpadded}[bl][t]}$
    else
      for $t := 0$ **to** T **do**
        $lab[label[i]][t] = lab[label[i]][t] + ab[s][t]$
        $ab[s][t] = \frac{ab[s][t]}{out_{unpadded}[label[i]][t]}$
  $lh = zeros(T)$
  for $t := 0$ **to** T **do**
    $lh[t] = \sum_{s=1}^{S} ab[s][t]$
  $loss = -\sum_{t=1}^{T} \ln lh[t]$
  $softmaxGrad = zeros(|\hat{A}|, (\bar{W}_{padded}/4))$
  for $t := 0$ **to** T **do**
    for $i := 0$ **to** $|\hat{A}|$ **do**
      $softmaxGrad[i][t] = out_{unpadded}[i][t] - \frac{lab[i][t]}{out_{unpadded}[i][t] * lh[t]}$
  return *loss*, *softmaxGrad*

Text Recognition

Декодирование последовательности

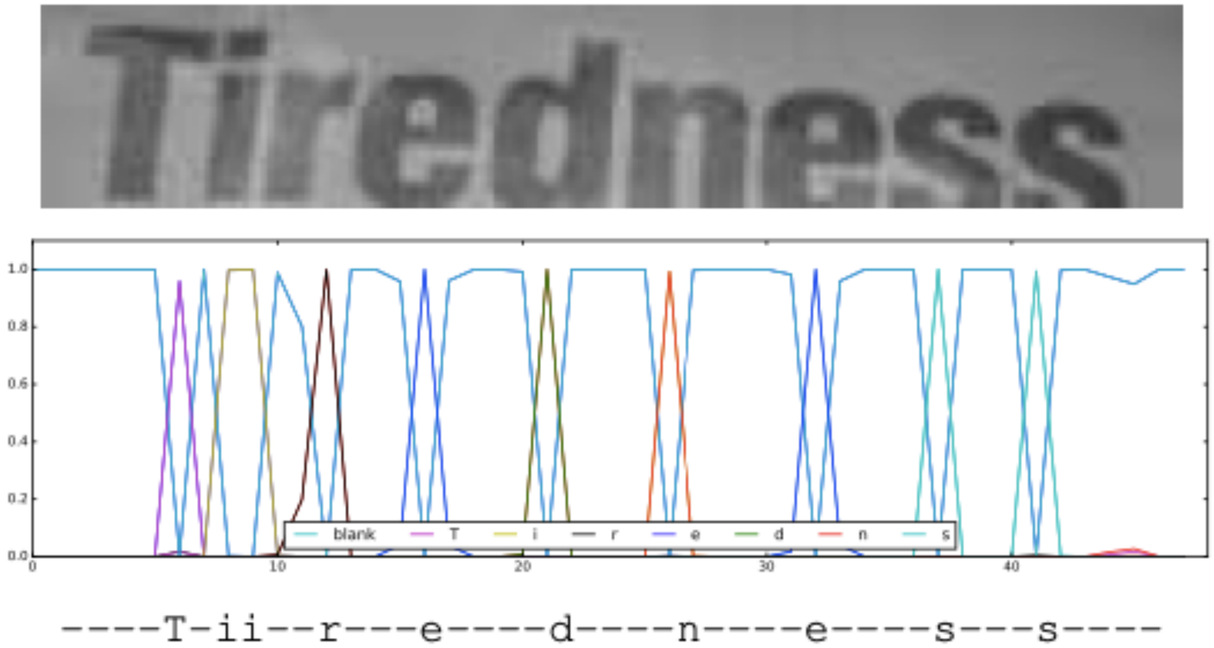
Реализовать декодирование:

1) На каждом отсчете t брать argmax(out[t]) и получить, например:

- - -MM- -A-ШШШ- -AA- -

2) Убрать из последовательности повторяющиеся буквы

3) Убрать пробелы



Deep TextSpotter: An End-to-End Trainable Scene Text Localization and Recognition Framework

Text Recognition

Подсчет точности

Реализовать:

1) Чтобы сравнить gt слово и decoded слово, используем расстояние Левенштейна: https://tirinox.ru/levenstein-python/

2) Полученное целое число делим на dist = max(len(gt), len(decoded)). Получаем 1.0, если последовательности совсем непохожи, и 0.0, если идентичны

3) Для каждой пары (gt, decoded) валидационной выборки считаем acc = 1 - dist

4) Все значения acc, полученные на 3 шаге, суммируем и делим на количество пар

P.S. Код подсчета расстояния Левенштейна можно взять из интернета :)

Дедлайн 20.10.2022 00:00