Feature extraction and raw feature selection

2022年4月1日　15:35

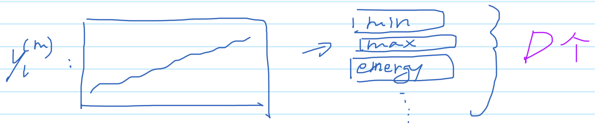
① feature extraction

using the "tsfresh" pack to obtain the raw feature map

for a history unit l $l=1,2,3\cdots L$, it has sensor $1,2,3,\cdots M$

take the signal of sensor m of unit l as $y_l^{(m)}$ → $1\times n_l$

the "tsfresh" can obtain lots of feature of $y_l^{(m)}$



therefore, after feature extraction, for a unit l, we have $D*m$ 个 feature

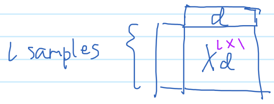
② raw feature selection

We have these propties i) the feature can give the hint to the target (?)

ii) the selected feature can distinguish the different kinds of signals

iii) the number of selected feature should be controlled

i) $\boxed{\text{RUL recognize part}}$. $Y_1 \to$ RUL, $X_d \to$ the feature d of all samples



assume the $Y_1 \sim N(\hat{Y}_1, \hat{\sigma}_{Y_1})$, $X_d \sim N(\hat{X}_d, \hat{\sigma}_{X_d})$

the mutual information $I(X_d, Y_1) = h(X_d) + h(Y_1) - h(X_d, Y_1)$ ↘ 联合熵

$$= \frac{1}{2}\log\left(\frac{\det \hat{\sigma}_{Y_1} \cdot \det \hat{\sigma}_{X_d}}{\det(\Sigma)}\right)$$ → covariance mat.

$\boxed{\text{mode recognize part}}$ $Y_2 \to$ mode $X_d \to \cdots$

$Y_2 \in [0,1,2,\cdots k]$ $I(X_d, Y_2) = h(X_d) - h(X_d \mid Y_2)$ $\to \sum_{k=1}^{K} P(Y_2=k)\, h(X_d \mid Y_2 = k)$

$P(X_d \mid Y_2 = k) \sim N(\hat{X}_d^{(k)}, \hat{\sigma}_{X_d}^{(k)})$

ii) roughly devide the training data into several ~~pear~~ kinds based on the RUL level and modes, assume that we have j parts

a feature d will have cov between j parts $\Rightarrow$ $V_{X_d} = \mathrm{Var}(X_d) - \sum_j \mathrm{Var}(X_d^{(j)})$

iii) denote the number selected feature is 100

therefore the feature seaction $\Rightarrow$

$$\max\ \gamma_1 \sum_d I(X_d, Y_1) + \gamma_2 \sum_d I(X_d, Y_2) + \gamma_3 \sum_d V_{X_d}$$

$$\sum_d X_d = S$$

$$X_d = 0 \text{ or } 1$$

其实就是取最大的 100 个⋯

$$\boxed{\begin{gathered}\max\ \gamma_1 \sum_d I(X_d, Y_1) + \gamma_2 \sum_d I(X_d, Y_2) + \gamma_3 \sum_d V_{X_d} \\ \sum_d x_d = S \\ X_d = 0 \text{ or } 1\end{gathered}}$$

其实就是取最大的100个…

the selected feature 1, 2, … S and we have the $\boxed{L \times S}$ mat.

$$\underbrace{L}_{\text{sample}} \times \underbrace{S}_{\text{feature}}$$

特征增强

2022年4月1日 16:29

this is a weird part

可能的解释

假设有两个 feature 1, feature 2

可能的解释 (in-service)

Runtime 小的序列不能很好的表现特征

⇒ 借用含有完整衰退过程的数据来表示

⇒ 将训练集的特征加权求和，越近的权重越高

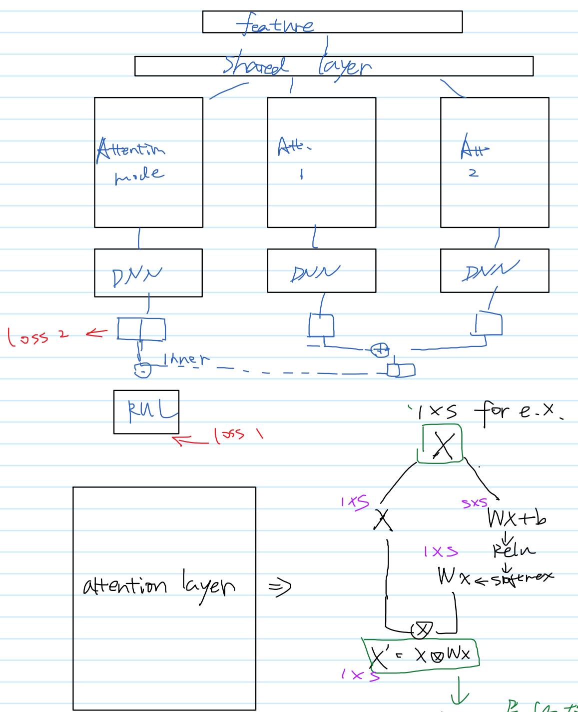
⇒ $\text{Weight} = \dfrac{1/\text{distance}}{\sum 1/\text{distance}}$ or $\text{weight} = \dfrac{\exp(-\text{distance})}{\sum \exp(-\text{distance})}$

按理来说②更 make sense，反而①的效果更好。

构建feature和RUL的关系

2022年4月1日　　16:39

建立神经网络　　　　　符号表示　　⊙ 内积　⊕ 拼接　⊗ 哈达玛积

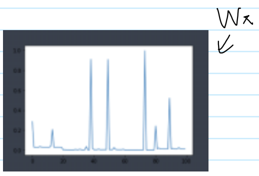


→ 然而实验表明 shared layer 还是不加的好

→ 也可多加几层，实验表明这样最好

Wx 在 Attention中记是　　　　，V 记是X

attention 层的输出可以是 X'，也可以是 X⊕X'



这里还用了个小改进，由于单独算的话大部分feature的权重都是0，所以拆分成很多个head，比如100个特征拆成5个大head，每个20个，分开计算权重然后再拼接在一起，这是我理解的multi-head attention的应用。虽然这个权重的方法看起来很奇怪，但是确实有些论文是这样处理的，还有通道注意力也使用了类似的方法，一个图片有很多通道，通过这样的方式获取权重。

效果展示　　RUL-level　　error　　std

```
matrix([[2.50000000e+01, 1.99635571e-02, 5.89919628e-03],
        [5.00000000e+01, 2.19749984e-02, 4.33802086e-03],
        [7.50000000e+01, 5.09131713e-02, 1.31361999e-02],
        [1.00000000e+02, 7.17816508e-02, 1.32667151e-02],
        [1.25000000e+02, 9.70969080e-02, 1.67776799e-02],
        [1.50000000e+02, 1.13511552e-01, 1.72295551e-02]])
```

这是没有特征增强的，对于RUL大的误差还是挺大的，我猜测可能和截断获取训练样本这样的方法有关，由于这个是非参数方法获得的特征，所以训练样本的分布方式会对结果产生比较大的影响，这个可能可以通过改进取样方式解决，但是这点我非常不确定

```
matrix([[2.50000000e+01, 1.99635571e-02, 5.89919628e-03],
        [5.00000000e+01, 2.19749984e-02, 4.33802086e-03],
        [7.50000000e+01, 4.87135107e-02, 1.22891108e-02],
        [1.00000000e+02, 4.94432974e-02, 9.29555482e-03],
        [1.25000000e+02, 6.34846845e-02, 8.76490113e-03],
        [1.50000000e+02, 7.89745741e-02, 9.03031720e-03]])
```

然后我就用了特征增强的方法魔改了好久，然后就获得了这样的效果。。。