CSLT 实习生考察报告

——机器翻译

2018.7.22

姜修齐

北京邮电大学本科2015级

联系方式
mail: xiuqi6666@gmail.com
tel: 18136626688

任务一　阅读机器翻译经典论文

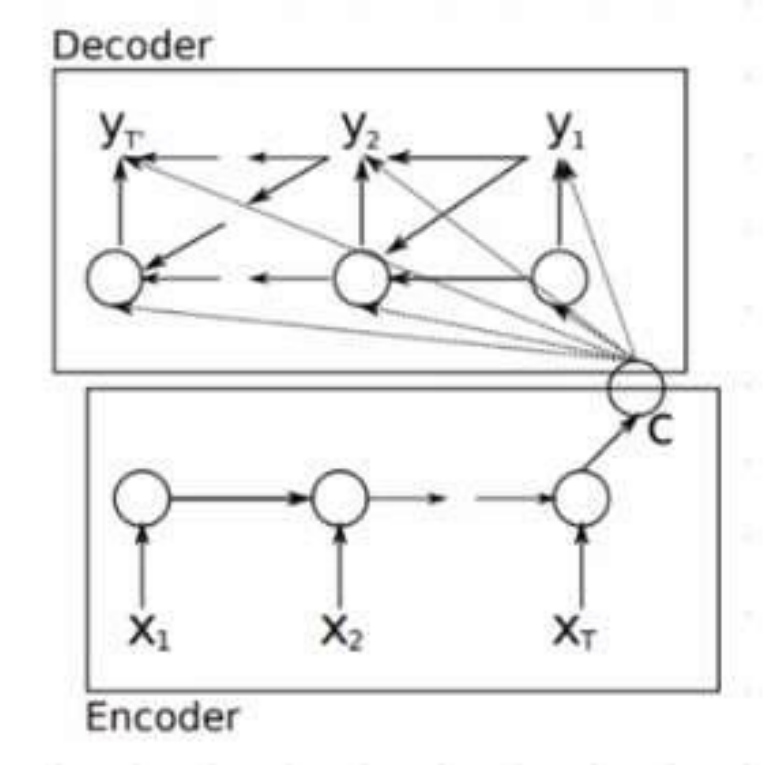
Neural machine translation by jointly learning to align and translate

"In this paper, we conjecture that the use of a **fixed-length vector** is a bottleneck in improving the performance of this basic **encoder–decoder architecture**, and propose to extend this by allowing a model to **automatically (soft-)search** for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly."

目前解决NMT问题的做法主要是基于encoder-decoder框架。但是，encoder部分把输入信息压缩到一个固定长度的vector中，这造成了性能的瓶颈。
这篇论文提出的模型就是在翻译的过程中自动在输入中寻找与输出目标有关系的部分帮助决策。

任务一　阅读机器翻译经典论文

Neural machine translation by jointly learning to align and translate

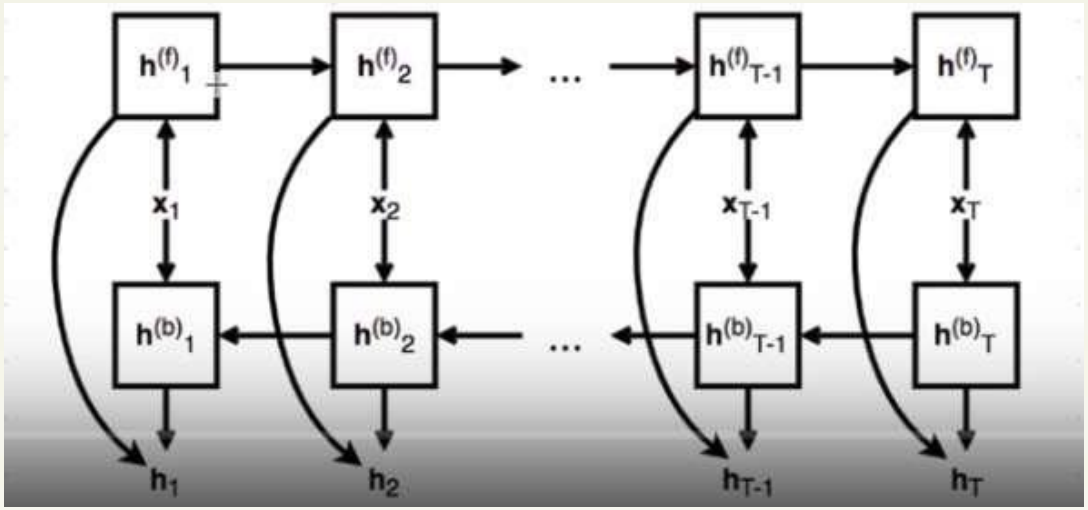


$$h_t = f\left(x_t, h_{t-1}\right)$$

$$c = q\left(\left\{h_1, \cdots, h_{T_x}\right\}\right)$$

任务一　阅读机器翻译经典论文

Neural machine translation by jointly learning to align and translate

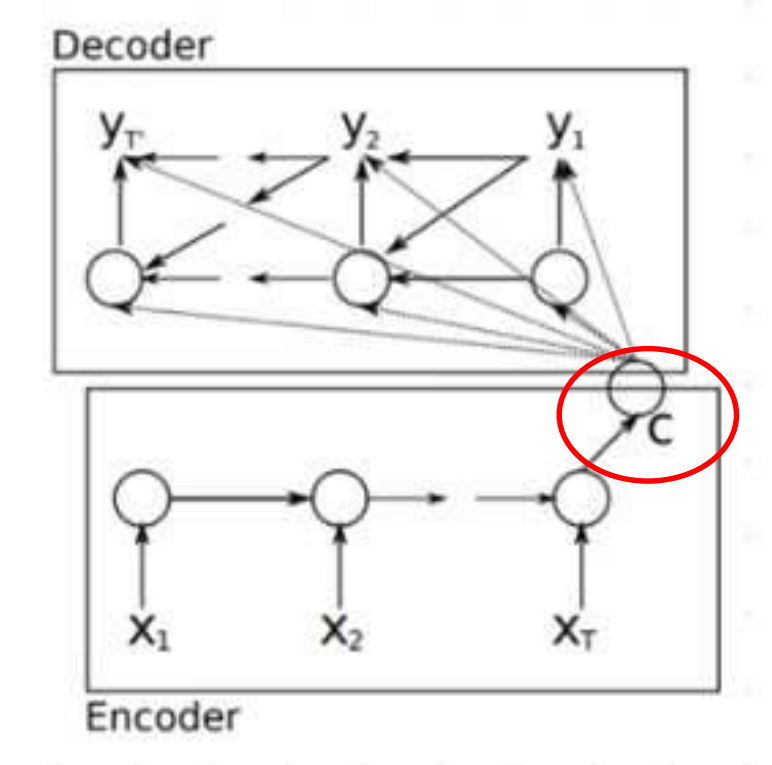


$$h_t = f\left(x_t, h_{t-1}\right)$$

$$c = q\left(\{h_1, \cdots, h_{T_x}\}\right)$$

任务一　阅读机器翻译经典论文

Neural machine translation by jointly learning to align and translate



$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c)$$

$$-\log \mathbb{P}(y_1, \ldots, y_{T'})$$

$$p(y_t \mid \{y_1, \cdots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c)$$

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

任务一　阅读机器翻译经典论文

Neural machine translation by jointly learning to align and translate

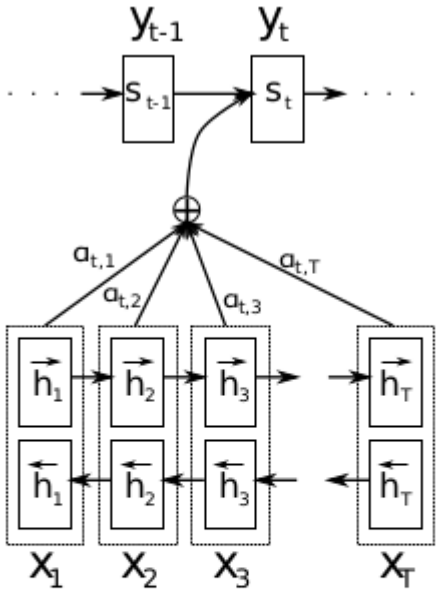


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

$$e_{ij} = a(s_{i-1}, h_j)$$

$$e_{ij} = v_a^\top \tanh(W_a s_{i-1} + U_a h_j)$$

任务二　阅读并应用机器翻译代码

https://github.com/CSLT-THU/CSLT_NMT

corpus: WMT2017 Parallel Data europarl-v7.cs-en

preparation : tokenisation & cleaning(80)

```
(tf1.0) [zhangsy@wolf08 CSLT_NMT]$ perl ./multi-bleu.perl data/test.trg < test.t
rans
BLEU = 27.01, 63.8/35.7/25.9/19.2 (BP=0.827, ratio=0.841, hyp_len=58, ref_len=69
)
```

任务二　阅读并应用机器翻译代码

corpus: WMT2017 Parallel Data europarl-v7.cs-en

https://github.com/CSLT-THU/CSLT_NMT

tf1.0 cs-en (hidden_units 200 hidden_edim 300 num_layers 1 batch_size 32)

	checkpoints	bleu(beam_size=10)	bleu(beam_size=20)
1	20000	15.77	15.93
2	40000	16.59	17.64
3	56000	20.36	20.12
4	64000	15.60	18.17
5	75000	17.77	17.92
6	80000	22.16	22.45

preparation	tokenisation & cleaning(80)
train_size	60000
dev_size	1000
test_size	1000
vocab.src	30000
vocab.trg	30000
初始化lr=0.00050000	
step56000时lr=0.00049500	purplex=5.64
step77000时lr=0.00049005	purplex=4.87

tf1.0 cs-en (hidden_units 200 hidden_edim 200 num_layers 1 batch_size 16)

checkpoints	bleu(beam_size=10)
100000	27.01

preparation	tokenisation
train_size	60000
dev_size	1000
test_size	1000
vocab.src	30000
vocab.trg	30000
初始化lr=0.00050000	

任务二　阅读并应用机器翻译代码

实验中遇到的问题和解决

OOM

数据集过大导致模型训练中出现OOM的问题。模型参数初始设定：

hidden_units 1000 -> 200

hidden_edim 620 -> 300

num_layers 1

batch_size 80 ->32

语料预处理

由于没有用中文，所以考虑不用分词，但是后来发现其他语种也需要预处理。（特殊符号 & 标点与字符的分隔）

tokenisation

truecasing

cleaning

数据集分割

分割数据集splitData.py

任务二　阅读并应用机器翻译代码

实验中遇到的问题和解决

```python
# -*- coding: utf-8 -*-
import os
import time


def mkSubFile(lines, srcName, sub):
    [des_filename, extname] = os.path.splitext(srcName)
    filename = des_filename + '_' + str(sub) + extname
    if sub == 1:
        dataname = 'train.' + datatype
    elif sub == 2:
        dataname = 'dev.' + datatype
    elif sub == 3:
        dataname = 'test.' + datatype
    else:
        dataname = 'useless_' + filename
    print('make file: %s as %s' % (filename, dataname))
    fout = open(dataname, 'w')
    try:
        fout.writelines(lines)
        return sub + 1
    finally:
        fout.close()
```

```python
def splitByLineCount(filename, *count):
    fin = open(filename, 'r')
    try:
        buf = []
        sub = 1
        i = 0
        for line in fin:
            buf.append(line)
            if sub == len(count)+1:
                continue
            if len(buf) == count[i]:
                sub = mkSubFile(buf, filename, sub)
                buf = []
                i += 1
        if len(buf) != 0:
            sub = mkSubFile(buf, filename, sub)
    finally:
        fin.close()


if __name__ == '__main__':
    begin = time.time()
    datatype = 'src'
    splitByLineCount('europarl-v7.cs-en.en', 200000, 1000, 1000)
    end = time.time()
    print('time is %d seconds ' % (end - begin))
```

任务二　阅读并应用机器翻译代码

实验

```
from 100 to 110
I tensorflow/core/common_runtime/gpu/pool_allocator.cc:247] PoolAllocator: After 5251
 get requests, put_count=5429 evicted_count=3000 eviction_rate=0.552588 and unsatisfi
ed allocation rate=0.540088
I tensorflow/core/common_runtime/gpu/pool_allocator.cc:259] Raising pool_size_limit_
from 160 to 176
I tensorflow/core/common_runtime/gpu/pool_allocator.cc:247] PoolAllocator: After 132
get requests, put_count=2153 evicted_count=2000 eviction_rate=0.928936 and unsatisfie
d allocation rate=0
I tensorflow/core/common_runtime/gpu/pool_allocator.cc:247] PoolAllocator: After 5582
 get requests, put_count=7855 evicted_count=5000 eviction_rate=0.636537 and unsatisfi
ed allocation rate=0.493013
I tensorflow/core/common_runtime/gpu/pool_allocator.cc:259] Raising pool_size_limit_
from 281 to 309
I tensorflow/core/common_runtime/gpu/pool_allocator.cc:247] PoolAllocator: After 3837
 get requests, put_count=7267 evicted_count=5000 eviction_rate=0.688042 and unsatisfi
ed allocation rate=0.417774
I tensorflow/core/common_runtime/gpu/pool_allocator.cc:259] Raising pool_size_limit_
from 372 to 409
I tensorflow/core/common_runtime/gpu/pool_allocator.cc:247] PoolAllocator: After 7127
 get requests, put_count=6666 evicted_count=3000 eviction_rate=0.450045 and unsatisfi
ed allocation rate=0.492493
```

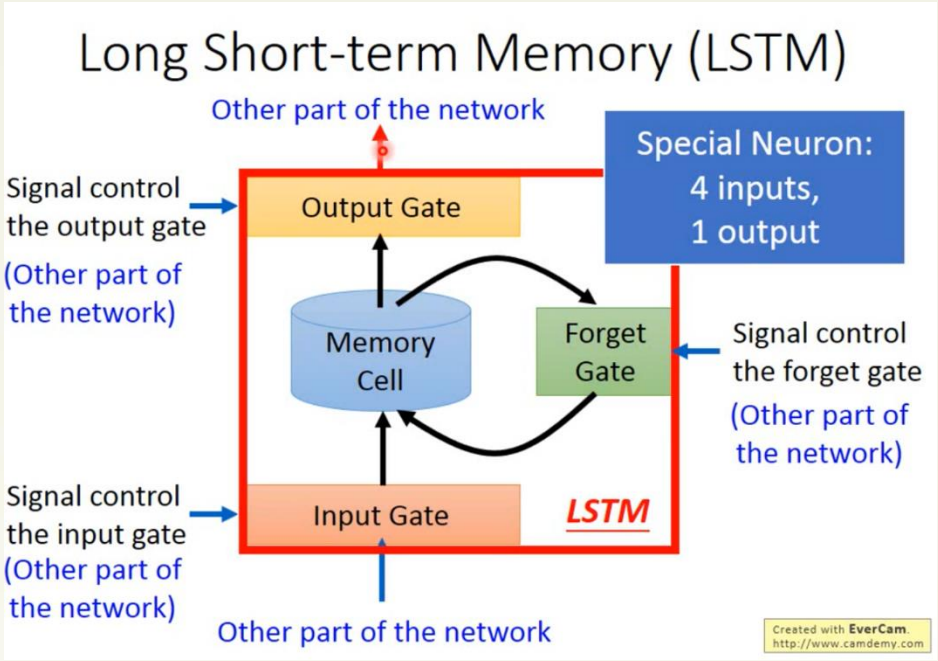
使用s
训练的
窗口或
会中断

在用户
/root/
加入：ServerAliveInterval 60

ssh客户端会每隔一段时间自动与ssh服务器通信一次，所以长时间操作不会断开。

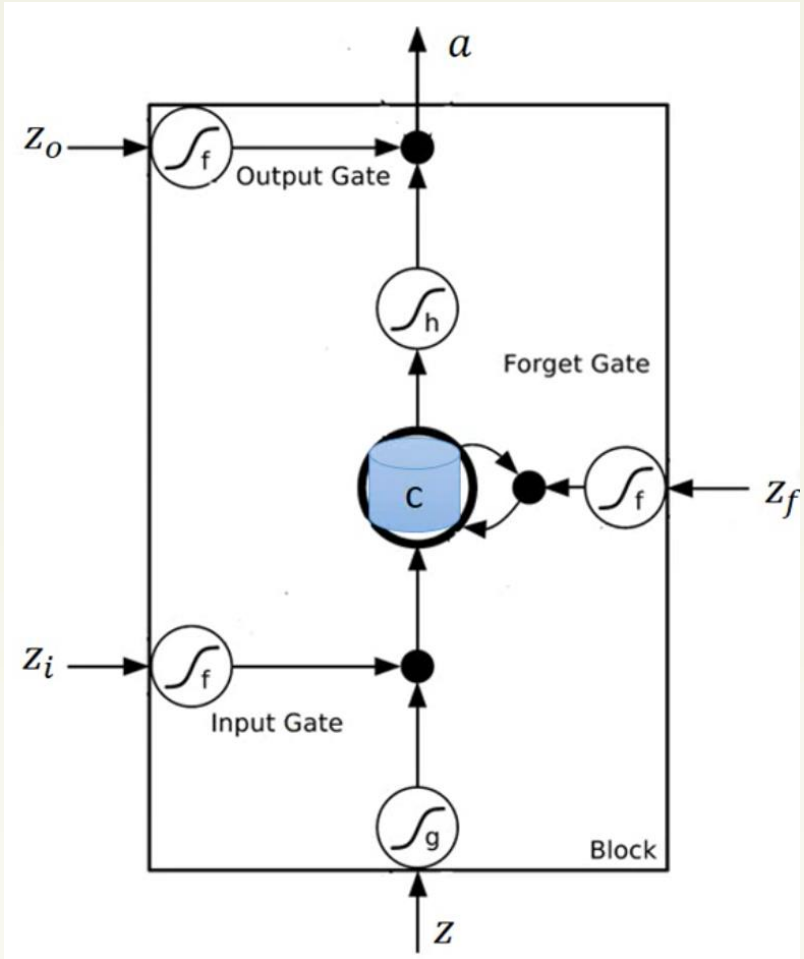
问题，但
tran没有
了几行

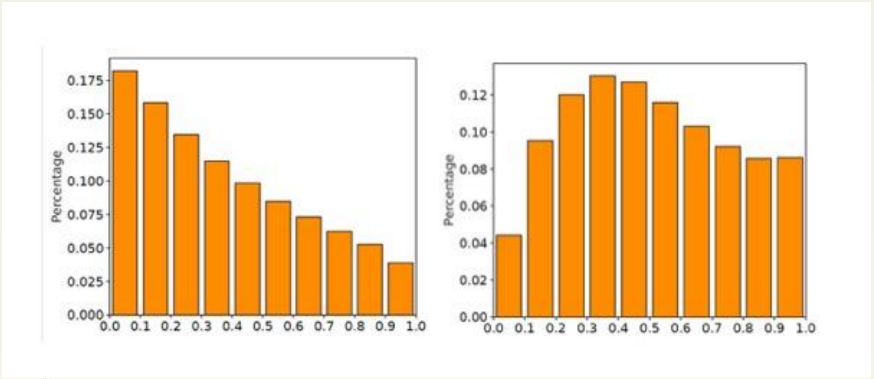
任务三　思考模型优化空间



长短时记忆网络（LSTM）是序列建模中被广泛使用的循环结构，可以防止普通RNN中出现的梯度爆炸和梯度消失问题。LSTM利用门结构来控制模型中信息的传输量。但是在实际操作中，LSTM的门通常处于半开半闭的状态，也就是说没有有效地控制信息的记忆与遗忘。我的试想是通过让模型的门接近“二值化”，可以更准确地去除或者增加信息，进而提高decode翻译部分的准确性。

任务三　思考模型优化空间

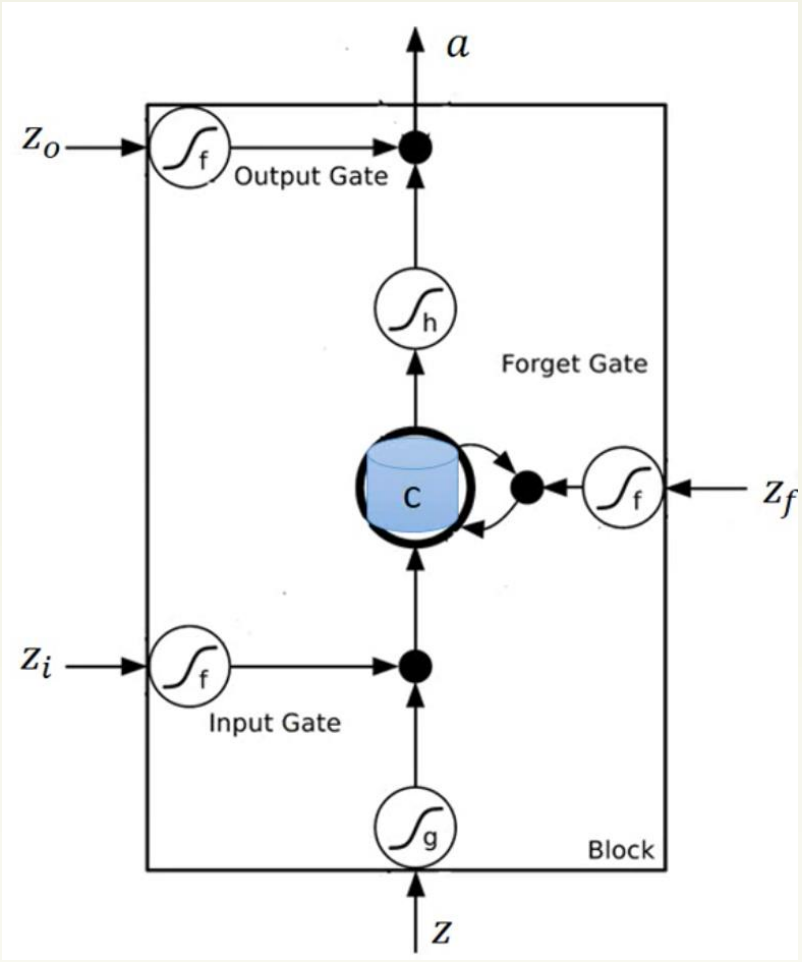


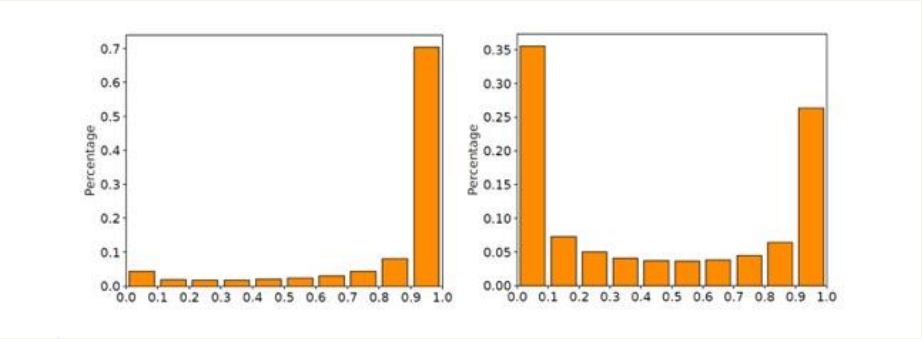


实际操作中，门是通过激活函数实现的：给定一个输入值**x**，通过sigmoid变换，可以转化成[0,1]值域内的值。在基于**LSTM**的端到端结构中，输入和遗忘门的输出取值往往在0.5左右，也就是说，**LSTM**中的门经常会处于半开半关的状态。

如果将门设置成二值化，可能会更加类似记忆的过程。

任务三　思考模型优化空间





Zhuohan Li, Di He, Fei Tian, Wei Chen, Tao Qin,Liwei Wang, and Tie-Yan Liu. "**Towards Binary-Valued Gates for Robust LSTM Training**." ICML 2018.

将门的输出“二值化”的最好办法是训练一个随机神经网络，其中门的输出是一个概率p，在伯努利分布中得到0/1的随机采样，借此去得到不同位置取0/1时的损失，进而优化参数得到最优p。

（Gumble-Gate LSTM）

谢谢！

联系方式
mail: xiuqi6666@gmail.com
tel: 18136626688