1. 现状总结

我们已经确定padding对ERNIE模型的性能提升，所以padding的PR需要合入Paddle的develop中。基于做了padding和测试时绑核的测试结果，Feiyue和Brian做了一个最新分析， 明确下一步的三个优化方向：

- QKV的合并
- Layernorm的并行化
- Remove reshape和transpose

Layernorm的并行化的PR已经merge，merge后Gaowei进行了最新的benchmark的测试，结果如下：

测试layernorm多线程结果#20895
ERNIE基准线是padding，绑核。

线程数	Ernie	Layernorm多线程	差别
单线程	251.734 ms	252.809ms	增加0.43%
20线程	36.17ms	32.9599ms	优化8.88%

layernorm OP 耗时

线程数	Layernorm耗时	Layernorm多线程耗时	差别
单线程	3510.71ms	3871.79ms	增加10.29%
20线程	3853.87ms	887.446ms	减少77.00%

做了Layernorm的并行化后，单线程上的运行速度微小变慢（0.43%）， 多线程上速度提升了8.88%。优化后的op hotspots如下图所示：

```
------------------------->     Profiling Report     <-------------------------

Place: CPU
Time unit: ms
Sorted by total time in descending order in the same thread

Event                          Calls       Total       Min.        Max.        Ave.         Ratio.
thread0::fc                    74000       24431.4     0.022329    10.0122     0.330154     0.740733
thread0::elementwise_add       38000       2194.6      0.033824    2.43725     0.0577527    0.0665378
thread0::transpose2            48000       1594.83     0.025024    8.91892     0.0332257    0.0483535
thread0::matmul                25000       1510.47     0.021713    2.56695     0.0604189    0.0457958
thread0::layer_norm            25000       894.7       0.03007     2.18405     0.035788     0.0271263
thread0::scale                 14000       765.974     0.004714    0.567349    0.0547124    0.0232235
thread0::softmax               12000       539.04      0.037764    0.522655    0.04492      0.0163431
thread0::reshape2              48000       357.825     0.005703    0.729517    0.00745469   0.0108489
thread0::load                  202         288.746     0.010017    181.046     1.42944      0.00875446
thread0::lookup_table          3000        257.236     0.071966    7.88935     0.0857453    0.0077991
thread0::stack                 1000        70.5179     0.064874    0.133298    0.0705179    0.00213802
thread0::tanh                  1000        48.1622     0.033202    8.11843     0.0481622    0.00146022
thread0::slice                 1000        11.4222     0.010438    0.034986    0.0114222    0.00034630
thread0::feed                  4000        10.9634     0.001066    0.022654    0.00274084   0.00033239
thread0::fetch                 1000        6.84497     0.005802    0.024523    0.00684497   0.00020753
```

接下来优化的方向为QKV的合并和Remove reshape和transpose。

2. 为什么要做QKV的合并和remove reshape和transpose

- QKV合并

在ERNIE中QKV的三个FC是分开做的，而在TF中，则0将QKV三个FC合成一个一个大的GEMM进行了运算。

Time(ms)	单线程		20线程	
	ERNIE	TF	ERNIE	TF
Q	1260	4310	140	277
K	1260		130	
V	1250		135	
Total time	3770	4310	405	277

根据比较一个attention module的QKV执行时间， 我们可以发现这种fusion在单线程的情况下比不fusion要慢540ms， 而在多线程的情况下比不fusion要快128ms。 我们使用MKL_VEBOSE统计出所有MKL GEMM的时间为下表

	20线程		ERNIE比TF慢
	ERNIE	TF	
12 layers total time(ms)	16008	14592	9.7%

这个9.7%的差距来自于QKV的fusion区别，由于这个只算了GEMM的时间，实际模型时间会更长，所以QKV在模型上带来的性能差距大概在6%-7%左右。

- 减少冗余的transpose2和reshape

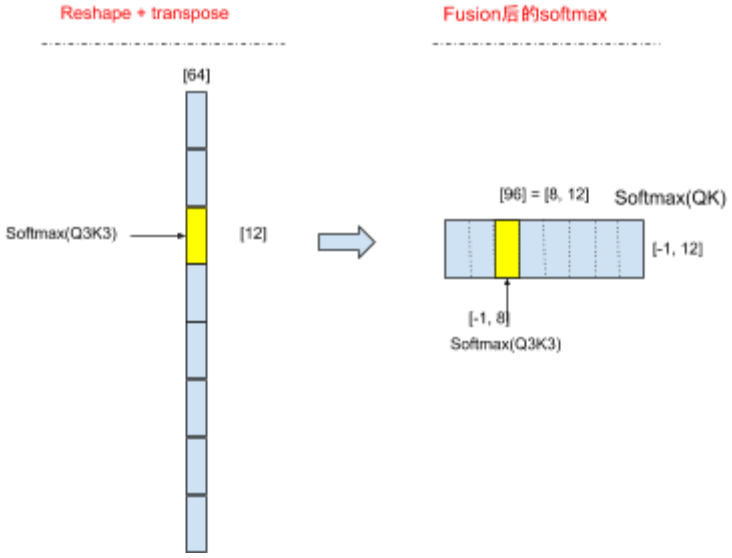
基于最新的profile的结果可知， 优化了layernorm的多线程的性能后，transpose2的时间占比在整体op时间范围内变成了第三位占比接近5%

```
thread0::transpose2         48000       1594.83     0.025024    8.91892     0.0332257   0.0483535
```

所以我们就要考虑之前讨论的remove transpose2和reshape2的PASS（PR20770）。这个PR针对BERT的优化做了三点特殊的处理：

1. Attention bias 的输入是已经经过transpose和reshape后的bias， 所以要对这个进行逆处理， 把“stack” op 换成“concat” op
2. Remove matmul前面/后面的reshape2和transpose2

3. 对除了最后一个softmax之外的，图中attetion module中的softmax都要设置以axis=1进行softmax, 这个修改就会非常依赖于图的结构，必须精确的找到每个attention module。

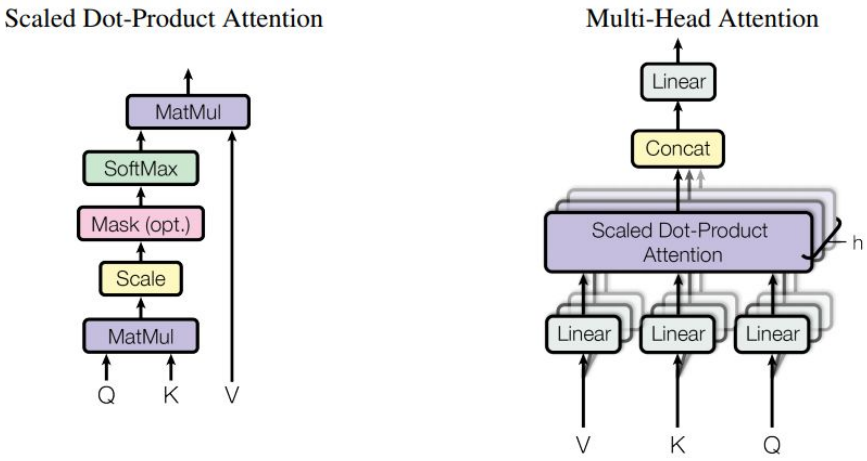


- Fusion该如何做

以上我们讨论了QKV合并和reshpe， transpose去掉的必要性，如果分别做这两个改图操作，我们会发现做出来的PASS通用性非常差，一旦图有一定的修改PASS就会失效，且QKV的合并将会是一个大的fusion，是目前Paddlepaddle不倾向的一种fusion。

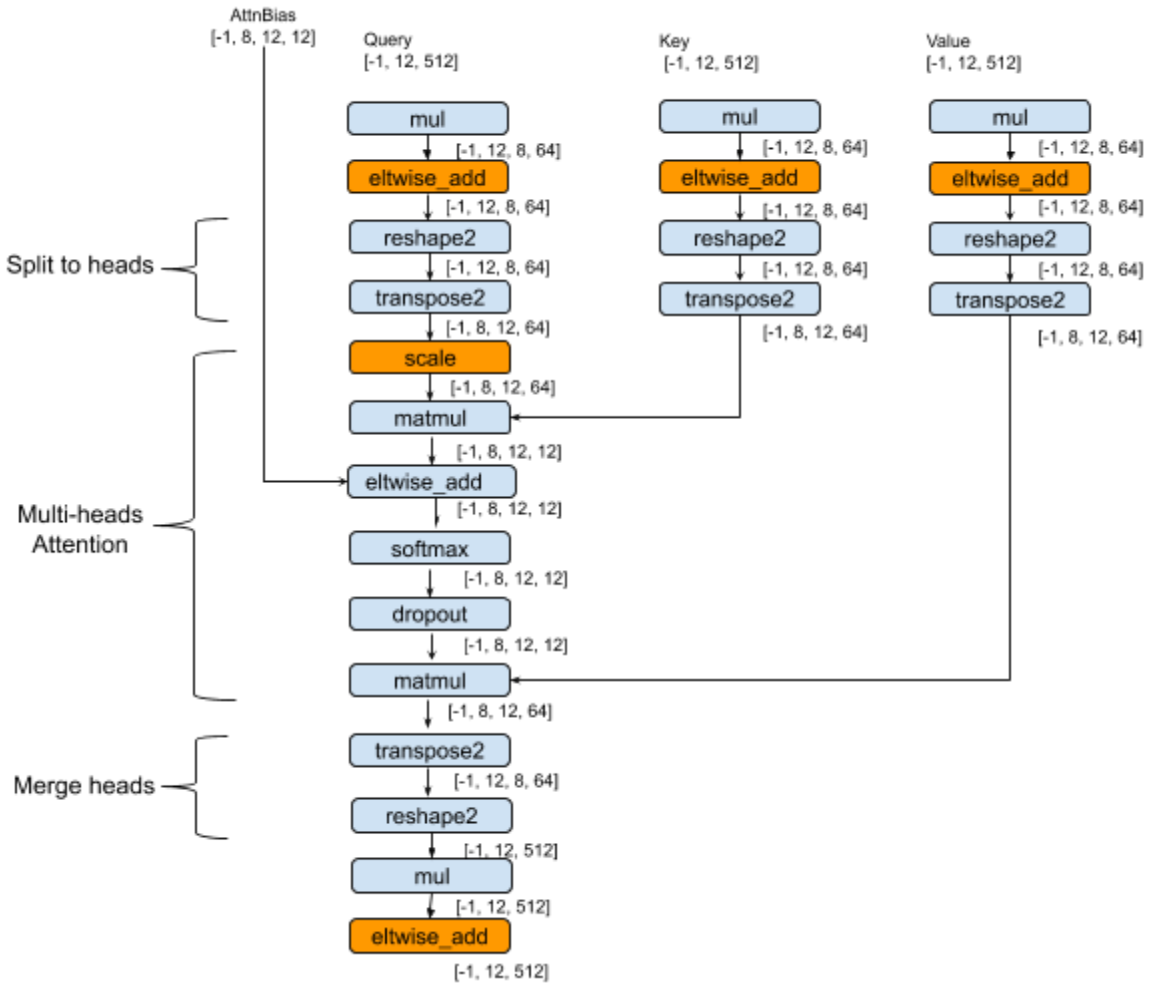
如果想比较通用的合并，我们建议直接创建一个attention module的op。现在Attention这个module被广泛应用于不同的NLP的模型中，就像当初的GRU一样，其实GRU也是由一组op组成的一个模块，只不过这个模块通用于不同的模型中，用户只需通过修改attribute就可以获得他们所期望的模块功能，所以我们来讨论一下，attention的op组合是否具有通用性和易用性。

➔ Attention module 的论文实现



➔ Attention module的Paddle中的op实现

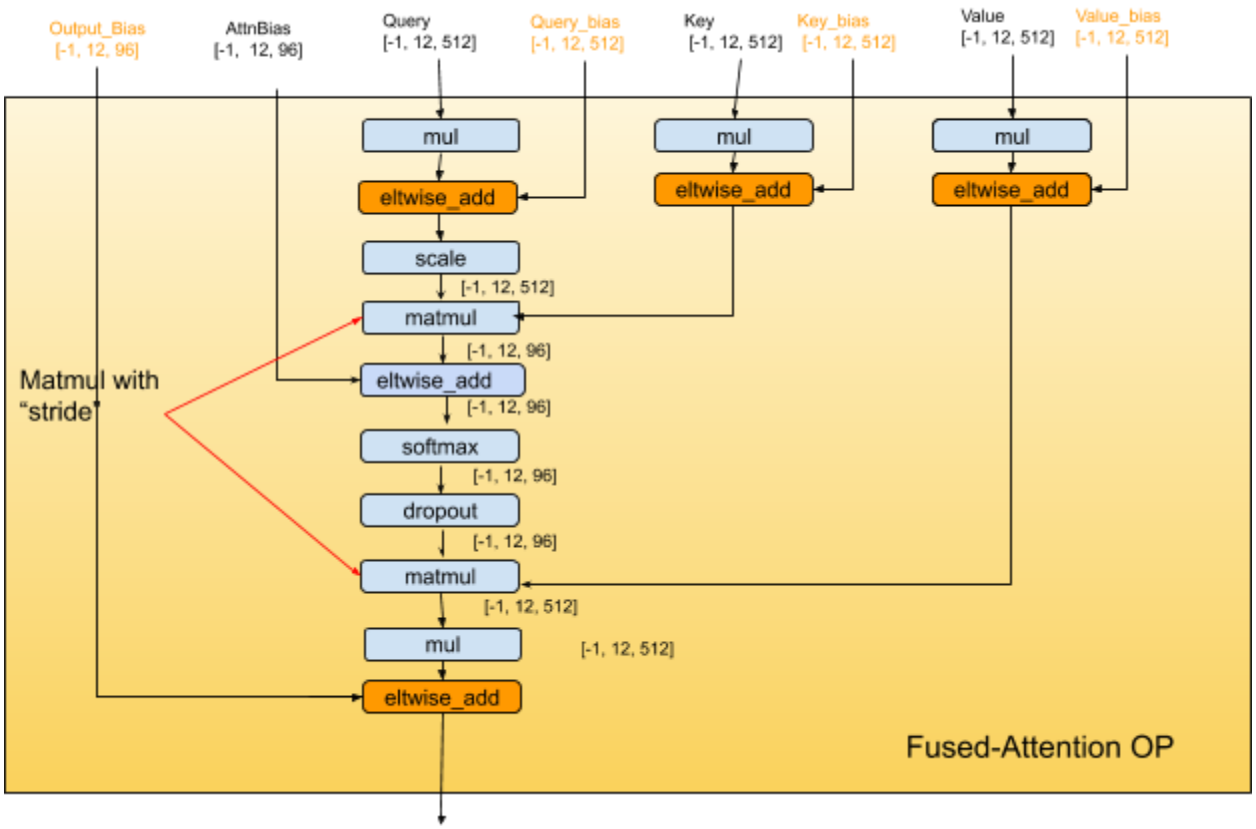
对比transformer的模型和bert的模型绘制出下图可以合成一个大的attention 模型的op组成。



图中标橙色的op为transformer和bert模型中关于attention module这个大fusion中不同的部分， 主要为两部分：

1. Mul是否带bias
    a. QKV的bias和output的bias，需要四个inputs传入
2. Query是否带scale
    a. 可以通过attribute传入

➔ Fusion之后的op为



- Pros and cons
  - 好处
    - 可以简单的做QKV的合并和移除不必要的reshape和transpose
    - ERNIE的20线程的性能预计提升~7%+~3%=~10%
    - 图会变得更简单
  - 坏处
    - 大op输入较多
    - 代码内部维护可能更复杂