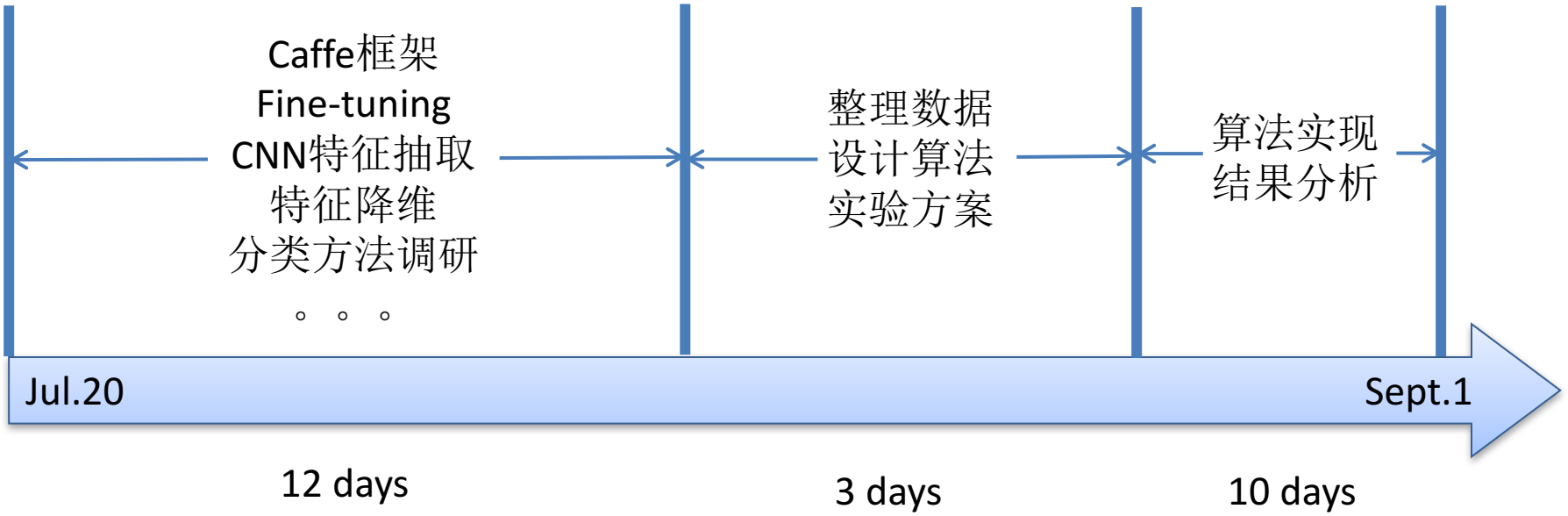
自动类目体系探索

张诚（九铸）
阿里巴巴 iDST
2015.9.1

Outline

- Background
- Approach
  - Preparation
  - Algorithm
- Experiments
- Conclusion

Timeline

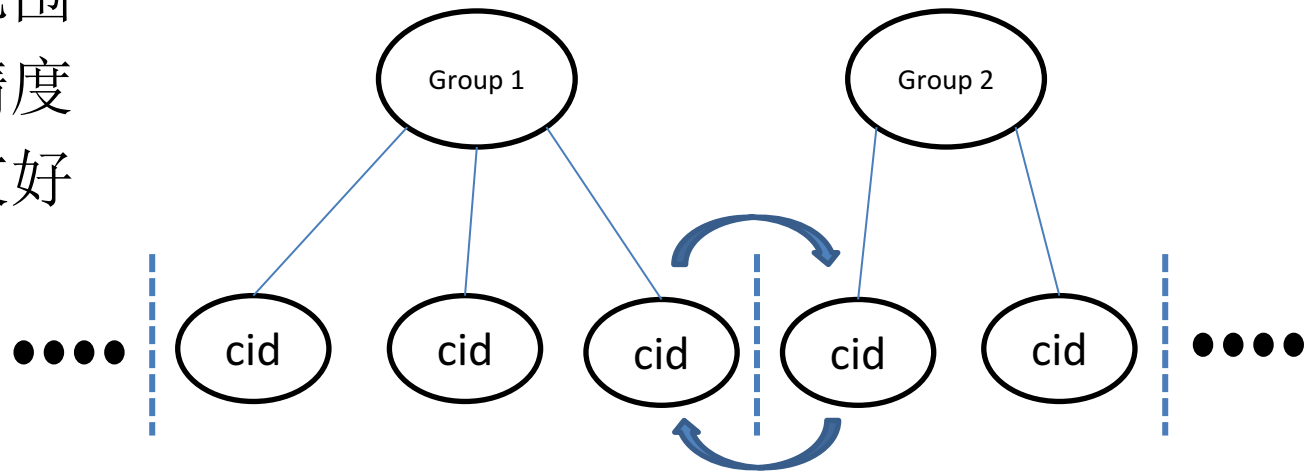

Caffe框架
Fine-tuning
CNN特征抽取
特征降维
分类方法调研
。 。 。
整理数据
设计算法
实验方案
算法实现
结果分析
Jul.20
Sept.1
12 days
3 days
10 days

Background

Motivation

- 全淘商品叶子类目数量繁多，体系不够合理，不利于图搜。需要形成拍立淘独有的类目中间层。为图搜提供良好的图像类目识别系统。
  - 对现有的叶子类目cid进行自动抽象，组合，浓缩
  - 手动调整费时费力
  - 缩小检索范围
  - 提高检索精度
  - 用户界面友好

……



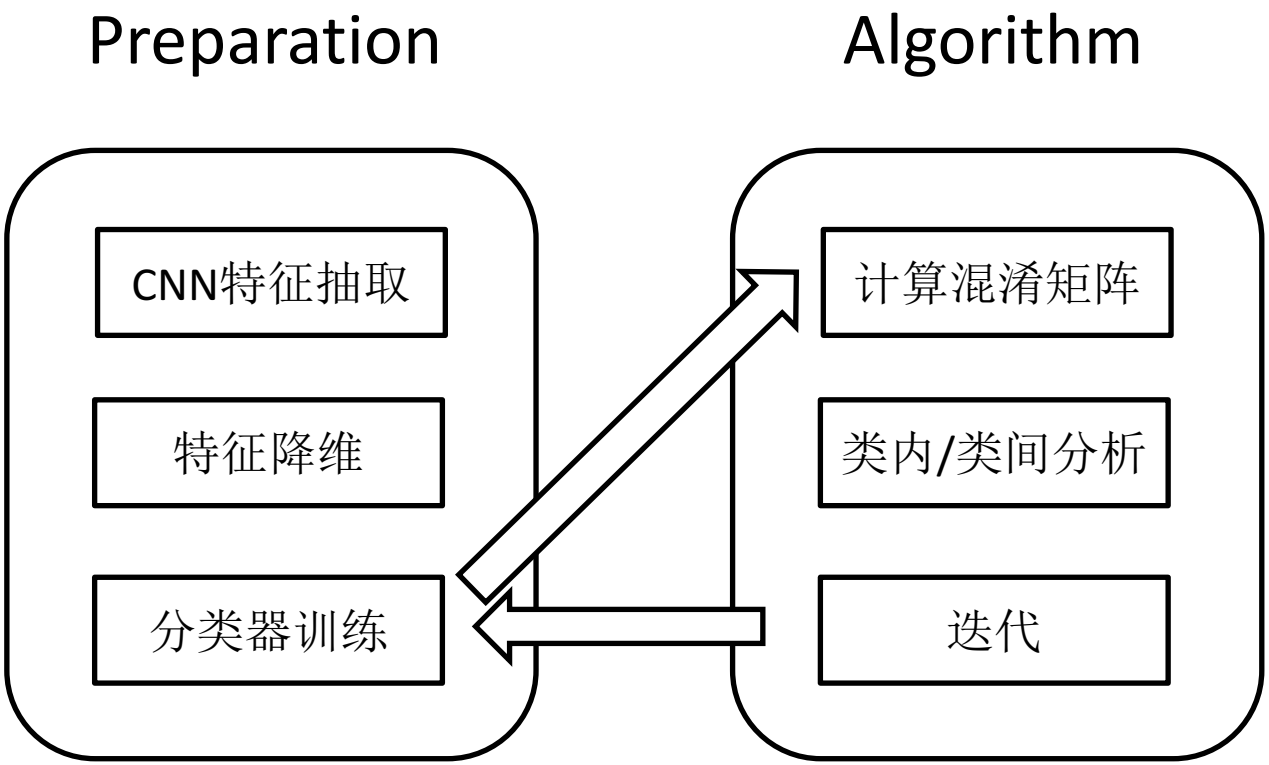
cited by mushi

Challenge

- 按视觉相似性重构商品类目体系
  - 无监督、自动化聚类可能产生不能理解的、非常复杂的语义节点
- 基于混淆矩阵的合并策略
  - 淘宝叶子类目是语义聚合的（外观形态差别可以非常大），而不是按照图像外观聚合的，很多叶子类目之间的相似性关系不容易自动化地衡量
  - 叶子类目的多样性：尤其是非重点类目
  - 训练和测试数据的异质性：组图训练，晒单图实拍图测试

Approach

Overview


Preparation
Algorithm
CNN特征抽取
特征降维
分类器训练
计算混淆矩阵
类内/类间分析
迭代

Preparation

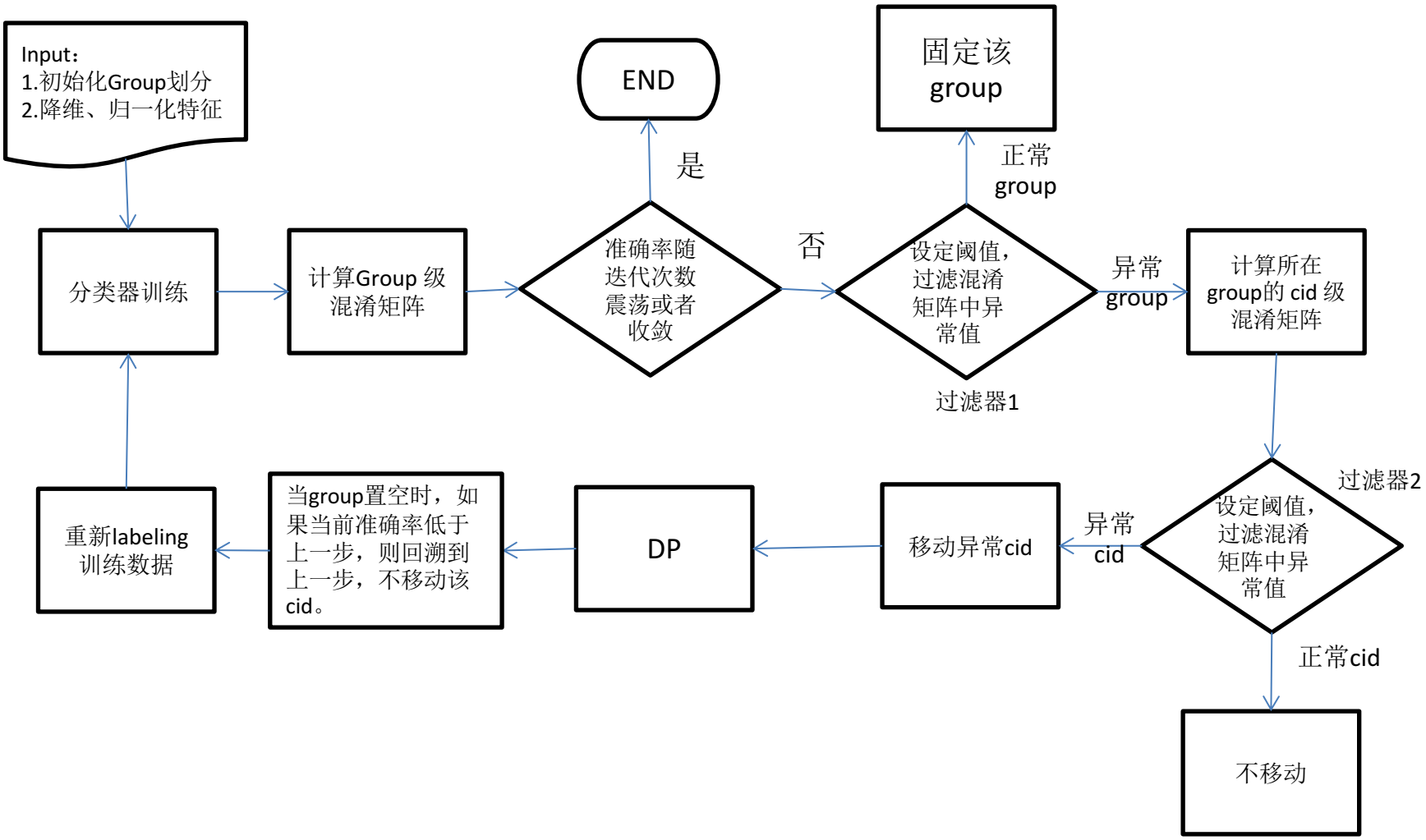
- 特征提取
  - 使用一个pretrained model提取训练数据和测试数据fc-6层的特征
  - 适当修改caffe数据层源码
- 特征降维
  - 降到多少维精度损失能够容忍，2048？1024？128？
  - 考虑速度
- 选择能够快速迭代的分类器
  - 考虑大数据、高维特征的适用性

Preparation (Cont.)

- 数据降维、训练时间及分类精度baseline
  - 结果表明，特征降维至1024精度损失不大，Liblinear分类器一次训练迭代时间可接受。

特征维度	groupNum	PCA降维时间 (350k数据，min)	一次迭代训练时间 (min)	训练方法	分类精度 (%)
128	13	<=15	<=1	Liblinear	33.9658
1024	13	<=30	<=5	Liblinear	41.7834
2048	13	120	<=10	Liblinear	42.2583
4096	13	-	20	Liblinear	42.8964
4096	13	-	-	CNN fine-tuning	48.9680
1024	187 （不合并）	<=30	<=30	Liblinear	26.9100

Algorithm flow



- Group级混淆矩阵

Group	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0.285018	0.009744	0.031669	0.023143	0.030451	0.037759	0.114495	0.035323	0.230207	0.079172	0.093788	0.019488	0.009744
2	0.053333	0.328889	0.064444	0.1	0.071111	0.013333	0.066667	0.042222	0.08	0.091111	0.02	0.055556	0.013333
3	0.039286	0.010204	0.528571	0.032143	0.029592	0.029592	0.103571	0.034694	0.063776	0.04898	0.018367	0.023469	0.037755
4	0.050239	0.011962	0.069378	0.660287	0.029904	0.016746	0.034689	0.010766	0.010766	0.038278	0.017943	0.039474	0.009569
5	0.05404	0.016469	0.089552	0.057128	0.164179	0.048893	0.222851	0.033968	0.071024	0.149768	0.027792	0.033968	0.030365
6	0.026432	0.008811	0.068722	0.017621	0.006167	0.58326	0.077533	0.021145	0.045815	0.098678	0.025551	0.008811	0.011454
7	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0.062617	0.026925	0.075141	0.030683	0.061365	0.03444	0.083281	0.36005	0.107076	0.088917	0.039449	0.018159	0.011897
9	0.093865	0.016256	0.033561	0.028317	0.031987	0.01311	0.075511	0.029365	0.528579	0.093865	0.031463	0.012061	0.012061
10	0.055661	0.013796	0.033302	0.03901	0.045195	0.037108	0.060894	0.014748	0.114177	0.52902	0.02236	0.011418	0.023311
11	0.101124	0.009631	0.030498	0.032103	0.020867	0.040128	0.077047	0.040128	0.150883	0.086677	0.370787	0.014446	0.025682
12	0.036728	0.023372	0.068447	0.175292	0.071786	0.013356	0.138564	0.033389	0.055092	0.055092	0.041736	0.27379	0.013356
13	0.069002	0.012739	0.052017	0.112527	0.035032	0.053079	0.113588	0.025478	0.055202	0.125265	0.096603	0.046709	0.20276

异常值　　对角线　　最高值

- Cid级混淆矩阵

groupID	cid	10	11	12	13	14	15	16	17	18	19	20	21	22
14	50009210	10	2	9	3	9	9	14	11	20	51	4	2	2
14	50013986	14	1	13	4	34	4	14	2	26	31	7	2	6
14	50013987	6	1	10	0	15	2	8	2	13	9	1	1	0
14	350706	3	0	6	10	0	0	5	0	4	19	1	0	0
14	50006279	11	0	24	8	32	0	35	0	5	8	14	4	17
14	210211	8	6	10	20	25	1	22	5	14	9	4	10	1
14	50010466	16	9	8	11	64	3	12	8	10	11	2	5	4
14	121408018	5	6	4	7	8	1	4	7	5	7	3	9	4
14	50001283	2	0	2	0	35	0	98	4	6	14	3	1	0
14	50001284	12	0	7	1	44	12	46	6	8	34	3	3	6
14	50011664	2	0	1	0	7	0	43	3	2	1	0	2	1
14	50006236	1	2	6	9	6	7	13	1	3	6	1	2	1
14	50004423	6	2	8	3	14	25	14	6	14	60	8	7	3
14	50026403	1	1	10	8	3	2	23	2	2	5	1	9	1
14	50025864	3	2	2	9	2	5	8	3	6	8	1	2	6
14	121368025	2	0	11	2	4	8	23	1	0	5	0	2	1
14	121456022	3	0	37	16	17	16	51	5	0	13	1	5	6
17	50012322	26	3	5	8	7	10	26	17	11	16	3	3	3
17	50012323	9	1	9	1	13	2	6	49	15	12	3	4	0
17	50012938	13	0	13	7	6	8	10	39	9	18	4	1	2
17	50012940	8	4	6	3	5	0	2	45	13	11	6	6	2
17	50013824	10	1	7	3	12	2	20	45	12	13	3	3	0
17	50013203	3	6	2	6	8	4	5	72	11	6	7	7	1
17	50017617	4	0	7	0	4	6	5	74	5	11	11	1	3
17	50017078	4	2	8	3	16	2	1	53	19	4	5	1	0
17	50017596	4	0	5	3	3	6	11	82	9	5	6	0	1
17	50017095	4	10	23	5	2	6	19	20	19	19	8	1	2
17	50017096	8	9	16	6	13	1	15	13	37	17	4	0	4
17	50018097	2	7	3	3	3	5	4	25	4	4	1	2	0
17	50017723	5	0	16	1	6	3	9	41	7	6	2	0	1
19	350201	3	7	4	3	8	5	14	4	23	146	0	1	8
19	50005000	1	0	2	3	0	0	3	0	7	28	1	0	0
19	50005002	4	2	3	2	1	5	3	0	6	26	2	0	0
19	350203	2	1	4	2	3	4	4	0	9	38	2	1	1
19	50011883	7	0	3	2	2	2	2	3	5	34	0	0	1
19	50008385	1	0	2	1	0	8	0	1	1	4	1	0	0

应该被分到的位置

最大数量位置

- 目前的方法

  Step 1：在group level的混淆矩阵上筛选出异常点。这个原则应尽量保证recall，所以将阈值设大一些是可以的，也是合理。

  Step 2：从Spt 1中包括的所有cid，计算混淆矩阵。筛选出问题cid，并移动。

Experiments

Data

- 初始化将187个cid按照语义人工为13个group，每个group下平均包含10+个cid，具体数据分布如下：

group	1	2	3	4	5	6	7	8	9	10	11	12	13	sum
train	22000	6000	40000	18000	32735	40000	41337	25102	42000	42701	18338	14257	14000	356470
test	821	450	1960	836	1943	1135	0	1597	1907	2102	623	599	942	14915

50022381	调羹/饭勺	10
50006943	饭盒/保温桶/保温提锅	10
50002798	碗	10
121456004	西餐餐叉	10
50024779	碟	10
50006759	水果叉/水果签	10
121454007	西餐勺	10
121418004	西餐刀具	10
50024780	盘	10
50002260	茶盘	10
50006782	咖啡勺	10

50024135	飞机模型	12
50024151	火车模型	12
50024150	船舶/舰艇	12
50024136	坦克/军事战车	12
50024166	军舰/航母	12
50136008	人偶/可动人偶	12
50008838	点读学习	12
50023774	智能公仔/娃娃	12
50006939	高达模型专区	12
203509	变形金刚模型专区	12
50018842	机器人/机甲成品	12
50005869	太空堡垒模型专区	12
201808	圣斗士星矢模型专区	12
50006804	魔方	12
50008829	彩泥/橡皮泥	12
251707	拼图/拼板	12
50023652	建构/拼插积木	12
122642005	奥特曼玩具专区	12
122692006	人偶/摆件/硬体公仔	12
122616007	海贼王手办	12

122846004	十字绣成品	21
50020803	苏绣	21
50021908	湘绣	21
50021909	蜀绣	21
50021910	粤绣	21
50024799	椅垫	21
50024801	蒲团	21
50024921	保健坐垫	21

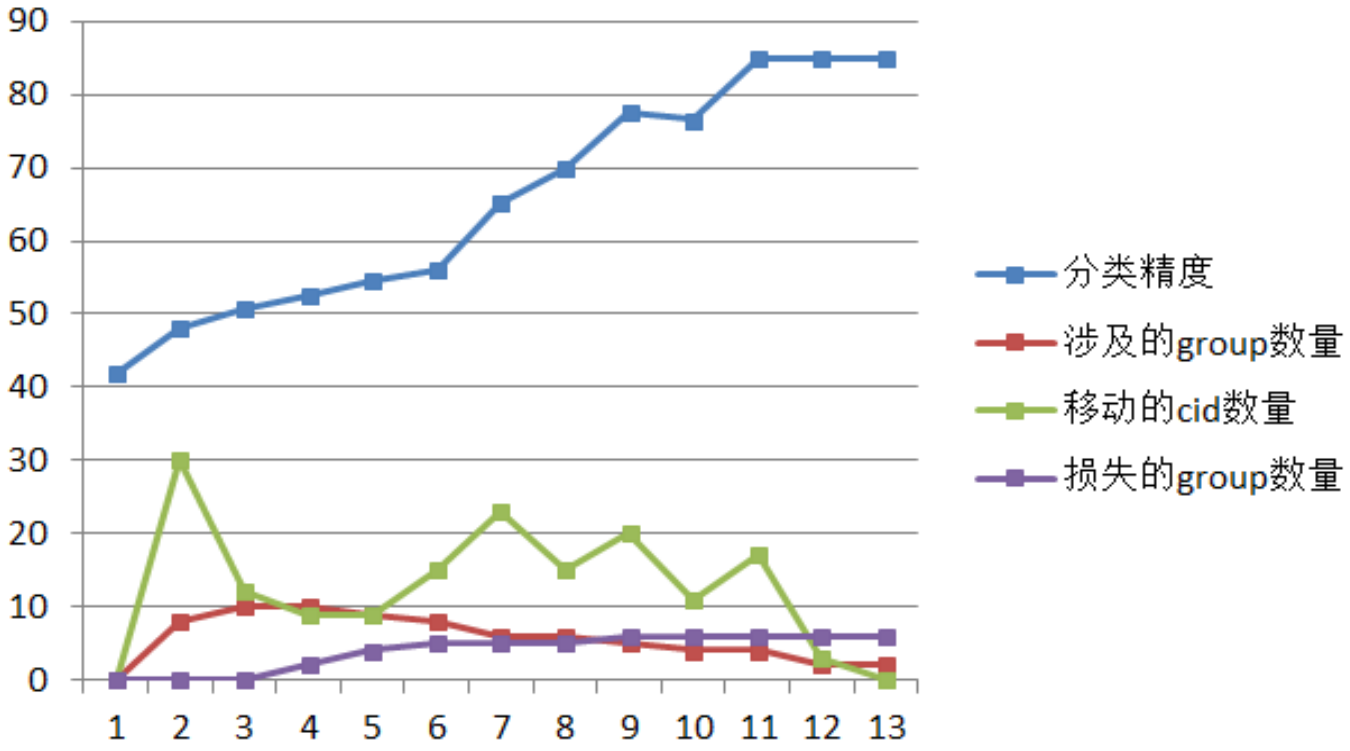
Results

- 迭代结果

Iteration	Error_group	Move_cid	Loss_group	Accuracy (%)
0	-	-	-	41.7834
1	8	30	0	48.1395
2	10	12	0	50.6269
3	10	9	2	52.5981
4	9	9	4	54.4821
5	8	15	5	55.9102
6	6	23	5	65.2363
7	6	15	5	69.7888
8	5	20	6	77.4992
9	4	11	6	76.5136
10	4	17	7	84.9883
11	2	3	7	84.9883
12	2	0	7	84.9883

Results

- 各参考量随迭代次数的趋势图



Case study

- 第五次迭代后group混淆矩阵

groupID	1	2	3	4	5	6	7	8	9
1	0.476657	0.125648	0	0.017867	0.196542	0.010375	0.126801	0.04438	0.001729
2	0.034648	0.661091	0.000398	0.017125	0.124652	0.014337	0.093987	0.051772	0.001991
3	0.060606	0.212121	0.21645	0.021645	0.181818	0.034632	0.220779	0.051948	0
4	0.054593	0.074523	0.000867	0.560659	0.098787	0.012998	0.111785	0.084056	0.001733
5	0.053416	0.12236	0.000621	0.036646	0.574534	0.01677	0.132919	0.058385	0.003106
6	0.049236	0.147708	0	0.03056	0.117148	0.380306	0.205433	0.067063	0.001698
7	0.029744	0.104104	0.001652	0.019829	0.108785	0.009915	0.646378	0.075737	0.00358
8	0.024127	0.120994	0.00036	0.045373	0.119914	0.009723	0.244869	0.432481	0.002161
9	0.022727	0.181818	0	0.011364	0.090909	0.011364	0.363636	0.079546	0.238636

Case study

- 前3轮迭代过程从语义上来比较合理，比如第3次迭代：

name	cid	所在组	语义	移动组	语义
调羹/饭勺	50022381	10	餐叉、条形	18	锅碗瓢盆
碟	50024779	10	餐叉、条形	18	锅碗瓢盆
西餐刀具	121418004	10	餐叉、条形	18	锅碗瓢盆
盘	50024780	10	餐叉、条形	16	摆件
咖啡勺	50006782	10	餐叉、条形	18	锅碗瓢盆
手套	50010410	11	帽子手套	13	枕套、方形
魔方	50006804	12	模型玩具	13	枕套、方形
羽毛球	50012322	10	餐叉、条形	16	摆件
旋转拖把	50003459	20	桶状物	18	锅碗瓢盆

Conclusion

- 尝试了基于混淆矩阵的分裂、合并策略在自动类目体系上的应用并探索了不同参数下的算法性能。
- 实验表明，该方法可以取得一些积极效果，自动类目体系构建是可行的。
- 前几轮迭代结果可以为分类提供有价值的参考并且节省一定人工耗时。

Future work

- 设计更为复杂的、合理的cid移动规则
  - 如果某个cid在当前group配置下，均匀分散在各个组，则该cid会拉低所有group的精度，可以考虑删除
  - 要check哪些cid在迭代过程中来回移动（冗余移动），这种情况可以考虑删除该cid
  - 每个Group至少保留一个cid，或者从其他group移动异常cid至该组
  - Group的自增长策略
  - 合理的迭代优化算法

- 准备更为均衡的训练-测试数据
  - 从cid级混淆矩阵可以观察，某些cid里只有几张或者几十张图。这种数据不足以支撑验证某条移动规则是否合理。
  - 训练-测试数据的平衡
  - 各个cid上数据的覆盖

一句话总结

修改caffe数据接口，基于提取的CNN特征，利用混淆矩阵的合并/分裂策略来进行类目体系优化。通过设计合理的规则来动态调整cid的归属组，达到了类目优化的目的。实验结果是积极的，我认为该方向也是可行的，但是还需要进一步根据实际业务设计合理的优化算法。

Thanks all !