基础思想

参考 adaboost 的思想，学习多个基分类器，最后集成一个大网络

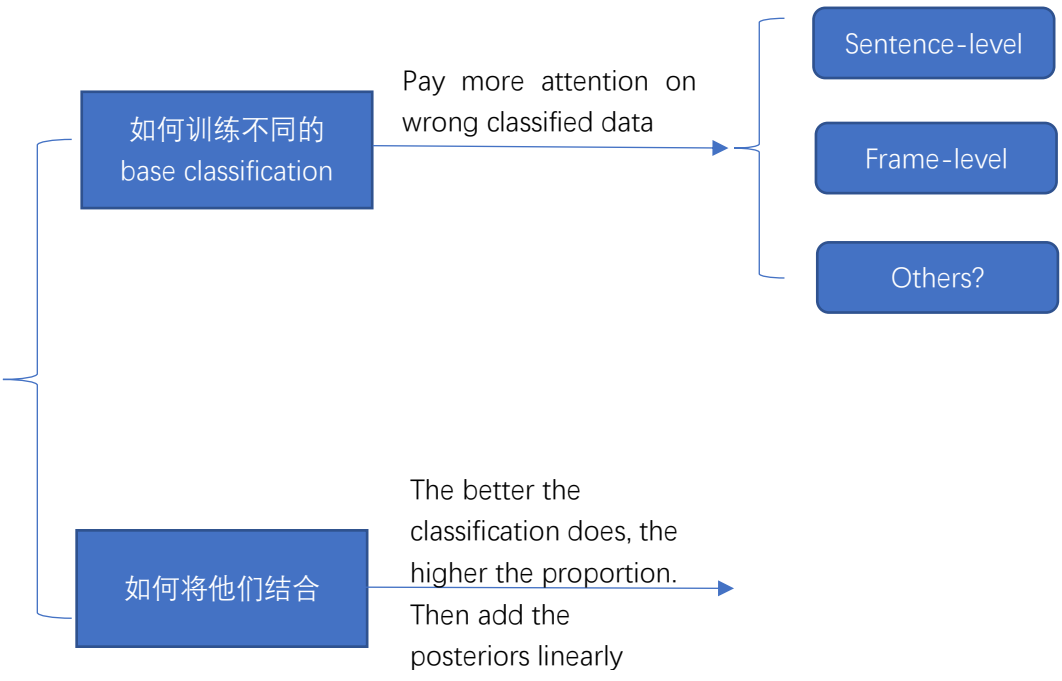
实验意义

1. 提高 LVCSR 任务的识别准确率
2. 提高解码速度

理论基础

1. Variance & bias
2. Adaboost 方法的泛化特性（不仅能在训练集上表现得好，测试集上也能有很好的表现效果）
3. Classification margin
4. 弱可学习=强可学习
5. Adaboost can do as well on NN as on decision tree.

思维导图

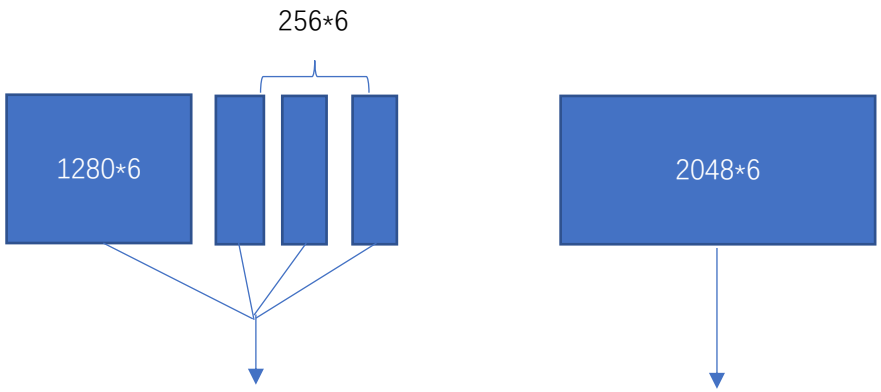


实验

实验 1

Experiment details

- Train data: WSJ / train_si284_sp_hires(+1 noise on all utterance)
  111954 utterance, about 240h
- Test data: WSJ / test_eval92 & dev93_sp_hires(+1 noise)
- NN_1(1280*6) NN_2/3/4(256*6)



Experiment procedure

对于第 n 个 NN

1. 对训练集的每个 utterance 解码，得到每个句子的 WER $W_n(i)$
2. 对每个句子的 WER 做归一化 $W_n(i)/\sum_i W_n(i)$，得到每个句子的概率$P_n(i)$
3. 用这个概率对原数据集进行 resample，保证数据集句子个数不变，得到新的训练数据集，用已有的 GMM-HMM 模型做 alignment，训练 NN。

训练好 n 个 NN 后，取测试集 WER 的倒数做归一化得到每个 NN 的比例，将 posterior 以此比例相加得到最终的 posterior 进行解码

results

	2048*6 (1280+256*3)	Ensemble 1+2+3+4	1280*6(1)	256*6 (2)	256*6 (3)	256*6(4)
bd_tgpr_dev93	**9.85**	10.53	10.37	17.21	18.80	20.68
bd_tgpr_dev93_fg	**8.95**	9.34	9.21	15.55	16.88	18.36
bd_tgpr_eval92	**6.31**	6.63	6.50	12.65	13.72	15.12
bd_tgpr_eval92_fg	**5.28**	**5.28**	5.46	10.77	12.26	13.22
tg_dev93	**11.63**	11.94	11.85	18.80	20.11	21.62
tg_eval92	**7.67**	8.40	8.22	14.11	15.17	16.39
tgpr_dev93	**12.38**	12.63	12.67	19.19	20.62	21.86
tgpr_eval92	**8.86**	**8.86**	**8.86**	14.74	15.91	17.21
Train_set			10.85	33.82	44.70	

Conclusion

1. It seems that ensemble model cannot do better than first 1280*6 model not to say 2048*6. maybe Adaboost can do better when NN have the same representation ability in other words these models have to have same structure.

2. The more complex data model learns on , the worse result they will have. That's as expected cause maybe harder training data really confused them.

3. When model trains on resampled data , they do better on test data. For example, model-2's WER on training data is 33.82% and model-3's WER on training data is 44.7%.This is really interesting cause model usually cannot do well on test set as they do on training set. Maybe they really learned some information.

4. The weight for each model is 0.86778 , 0.05940 , 0.04139, 0.03143 and they sum up to be 1.0 . I think it's more reasonable to weight the softmax layer before log. However , it do much better to directly weight the log-softmax layer which is unexpected.

存在的问题：

1. 原计划是训练几个大小相同的 NN，但是手误选错了第一个 NN 的模型结构，所以最后的结果是一个大网络带 3 小网络，这可能给网络带来信息不均衡的问题。
2. 训练集是在基础 WSJ 80h 训练集的基础上做了速度扰动扩充成 240h，训练集有点问题

实验 2

研究目的

分析了 priors 对结果的影响
分析了之前的训练方法无论是在噪音语音上还是在纯净语音上都没有变好的趋势

Experiment details

- Train data: WSJ / train_si284(+3 noise each with probability = 0.3)
  37318 utterance, about 80h
- Test data: WSJ / test_eval92 & dev93(+3 noise each with probability = 0.3)
- NN_1/2/3/4/5（256*6）

Experiment procedure

对于第 n 个 NN
4. 对训练集的每个 utterance 解码，得到每个句子的 WER $W_n(i)$
5. 对每个句子的 WER 做归一化 $W_n(i)/\sum_i W_n(i)$，得到每个句子的概率$P_n(i)$
6. 用这个概率对原数据集进行 resample，保证数据集句子个数不变，得到新的训练数据集，用已有的 GMM-HMM 模型做 alignment，训练 NN。

训练好 n 个 NN 后，取测试集 WER 的倒数做归一化得到每个 NN 的比例，将 posterior 以此比例相加得到最终的 posterior 进行解码

Results

basic results:

	1280*6	256*6 (1)	256*6 (2)	256*6 (3)	256*6(4)	256*6(5)
bd_tgpr_dev93	15.97	19.38	27.52	29.04		
bd_tgpr_dev93_fg	14.45	17.59	25.49	27.69		
bd_tgpr_eval92	13.20	16.89	22.24	23.57		
bd_tgpr_eval92_fg	11.89	15.59	20.26	22.08		
tg_dev93	17.40	20.03	28.11	30.58		
tg_eval92	14.05	17.92	23.25	24.61		
tgpr_dev93	18.00	21.11	28.59	31.33		
tgpr_eval92	14.78	19.01	23.91	25.64		
train set		19.83	46.87			

Ensemble result:

	256*6 (1)	**256*6 (2)**	**Ensemble(1)_(2)** weight1: 0.75595 weight2: 0.24405	**Ensemble(1)_(2)** weight1: 0.5647 weight2: 0.4353
bd_tgpr_dev93	19.38	27.52	20.66	27.41
bd_tgpr_dev93_fg	17.59	25.49	19.04	26.12
bd_tgpr_eval92	16.89	22.24	17.77	19.71
bd_tgpr_eval92_fg	15.59	20.26	16.23	18.36
tg_dev93	20.03	28.11	21.54	28.39
tg_eval92	17.92	23.25	18.18	20.95
tgpr_dev93	21.11	28.59	22.74	29.04
tgpr_eval92	19.01	23.91	19.19	21.76

conclusions

1. 显然 ensemble 的时候表现越好的 base classifier 权重越大，ensemble 的结果越好
2. 虽然ensemble的结果没有最好的classifier的结果好，但是ensemble绝对比每个classifier的 WER 按比例相加的结果好，具体如下

Test_set	1	2	3	4	5	6	7	8
Model(1)	19.38	17.59	16.89	15.59	20.03	17.92	21.11	19.01
Model(2) use prior of model(1)	53.50	52.87	44.16	43.45	55.00	45.95	54.48	45.47
Ensemble(1)_(2) weight1: 0.5647 weight2: 0.4353	27.41	26.12	19.71	18.36	28.39	20.95	29.04	21.76
Weight1*wer1+ weight2*wer2	34.23	32.95	28.76	27.72	35.25	30.12	35.64	30.53

Test_set	1	2	3	4	5	6	7	8
Model(1)	19.38	17.59	16.89	15.59	20.03	17.92	21.11	19.01
Model(2) use prior of model(1)	53.50	52.87	44.16	43.45	55.00	45.95	54.48	45.47
Ensemble(1)_(2) weight1: 0.75595 weight2: 0.24405	20.66	19.04	17.77	16.23	21.54	18.18	22.74	19.19
Weight1*wer1+ weight2*wer2	27.71	26.20	23.55	22.39	28.56	24.76	29.25	25.47

存在的问题

1. 训练第 2，3 个 classifier 的时候，对数据集进行了 resample，但是都用的是第一个训练集的 prior，做了一组对比实验，可以看出 prior 对模型的识别结果影响很大

Test_set	1	2	3	4	5	6	7	8
Model(2)	27.52	25.49	22.24	20.26	28.11	23.25	28.59	23.91
Model(2) use prior of model(1)	53.50	52.87	44.16	43.45	55.00	45.95	54.48	45.47

Further research of the result

1. WER on noise test set & clean test set

	1			2			3		
	total	clean	noise	total	clean	noise	total	clean	noise
bd_tgpr_dev93	19.38	8.57	24.41	27.52	9.73	35.87	29.04	10.99	37.47
bd_tgpr_dev93_fg	17.59	7.45	22.25	25.49	8.17	33.51	27.69	9.62	36.17
bd_tgpr_eval92	16.89	5.39	23.05	22.24	6.27	30.26	23.57	6.95	31.99
bd_tgpr_eval92_fg	15.59	4.30	21.32	20.26	4.72	27.92	22.08	5.65	30.18
tg_dev93	20.03	10.00	24.77	28.11	11.33	36.00	30.58	12.40	39.51
tg_eval92	17.92	7.05	23.18	23.25	6.84	31.50	24.61	7.36	33.36
tgpr_dev93	21.11	10.91	26.18	28.59	11.56	36.71	31.33	12.89	40.04
tgpr_eval92	19.01	7.21	24.91	23.91	7.98	31.96	25.64	8.76	33.74

2. The data composition of resampled data(the percentage of noise data)

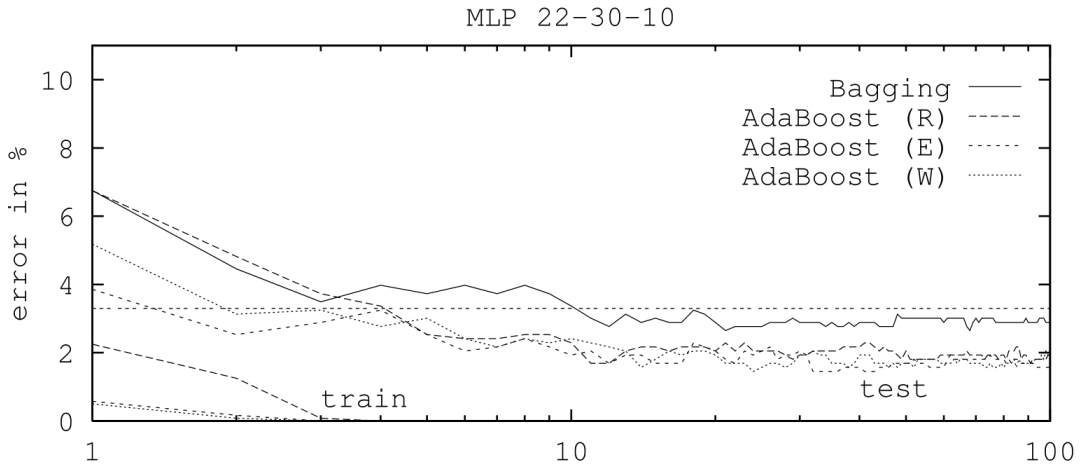
	Noise
train_set(original)	65.9%
train_set(resample_1)	80.7%
train_set(resample_2)	90.3%
test_dev93	67.0%
Test_eval92	63.3%

It seems that model trained on resampled data can neither do better on clean data nor on noise data. As for why this situation happens, to my opinion, maybe model concentrates on selected data instead of whole data information when it trains on resampled data. And I'm doubtful that when model performs worse can ensemble method works.

实验 3

探究 base classification 的个数对 ensemble 的结果有没有影响

参考论文的 error 趋势和 network 个数的关系



Experiment details

- train data
  data/train_si84 : 1 noise for every sentence. 7138 utterances , about 15h.
- test data
  data/test_dev93 & data/eval92 : 1 noise for every sentence. 503/333 utterances.
- Network 个数 5 个

Results

	GMM/tri4b_1_noise	256_1	256_2	256_3	256_4	256_5
bd_tgpr_dev93	17.84					
bd_tgpr_dev93_fg	15.75					
bd_tgpr_eval92	11.82					
bd_tgpr_eval92_fg	10.30					
tg_dev93	18.73					
tg_eval92	13.24					
tgpr_dev93	19.70					
tgpr_eval92	14.21					

Some results lost, finding…

Conclusions

Ensemble 的结果仍然没有 1 的好。

实验 4

- train data
  data/train_si84_half : 1 noise for every sentence. 3500 utterances , about 7h.
- test data
  data/test_dev93 & data/eval92 : 1 noise for every sentence. 503/333 utterances.

因为目前数据量只有 7h，同时考虑到论文中提到当神经网络节点数太多的时候不能发挥 adaboost 的效果，所以采用的神经网络结构为 3*100（不包含输入层和输出层）

目前的实验问题：

1. 如何解决 priors 的问题
   方案 1：把 priors 信息去掉，仅用神经网络得到的 posterior 解码。Kaldi 中的 priors 和 posterior 结合方式写死在解码过程中，ensemble 的时候无法用不同的 priors 解码
   方案 2：用 re-weight 而不是 re-sample，这样 priors 的信息一样。但是短期无法实现…
2. Re-sample 的方法
   反复看论文终于理解了为什么 re-sample 近似等于 re-weight。想要近似等于 re-weight，需要在每个 epoch 都 resample 一次，而且 epoch 数足够大。但是我在训练新的 base classifier 的时候只在最开始的时候 resample 了一次。这种方法虽然能让新的 NN 关注未学习好的数据集，但是不是对 re-weight 的近似。但论文中也用到了这种方法，论文中这种方法虽然效果不是最好，但是至少比单个的 NN 表现得好。

   并且我对每个句子计算的 re-sample 概率也有问题，单纯得用 WER 的归一进行 resample 不太合理，可能导致分类边界不明显。
3. Pdf 数过多
   ASR 的 pdf 数为 3400 左右，我现在实验的网络结构包含输入输出是 40-100-100-100-3400，论文中使用的网络最小也是 22-10-10。感觉 pdf 数也就是神经网络做分类任务的类别太大。
4. Frame-level or sentence-level
   虽然从 sentence-level 去 resample 数据集和 NN 在 frame-level 上做 classification 的原理相背离，但是 Kaldi 做 egs 的方法让 frame-level 的 re-sample 实现起来很困难。而且 frame-level 的数据量还是确实太大
5. 论文中提到 adaboost degrades a lot in the presence of significant amounts of noise。噪音语音数据集做实验是否合理?