信也科技杯 FINVOLUTION BIG DATA COMPETITION | 信也 FINVOLUTION

数智创新 声至未来

DEEP IN DIALECTS, FOR FUTURE WAVE

第八届信也科技杯算法大赛

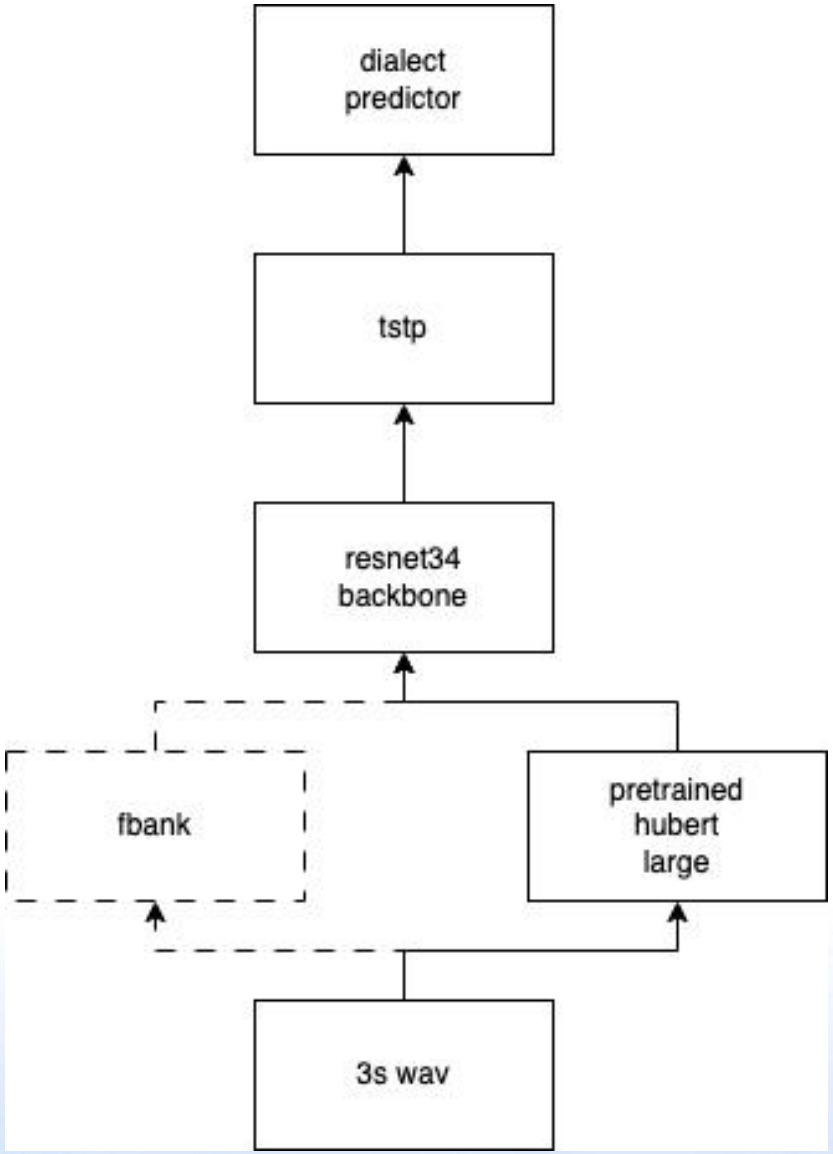
THE 8TH FINVOLUTION DATA SCIENCE COMPETITION

队伍:aluminumbox

目录

CONTENTS

初赛方案

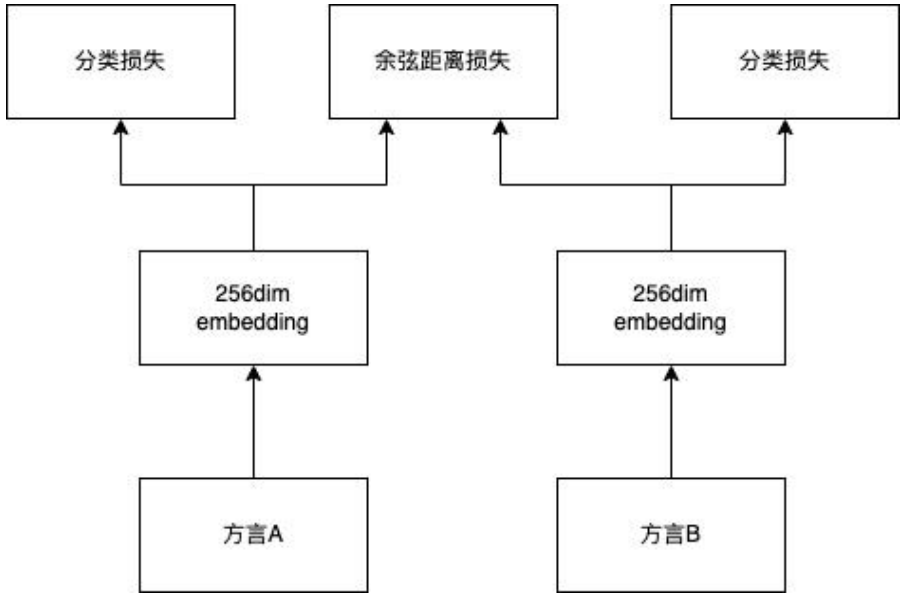


Layer name	Structure	Output
Input	–	$40 \times T \times 1$
Conv2D-1	3×3, Stride 1	$40 \times T \times 32$
ResNetBlock-1	$\begin{bmatrix} 3 \times 3, 32 \\ 3 \times 3, 32 \end{bmatrix} \times 3$, Stride 1	$40 \times T \times 32$
ResNetBlock-2	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 4$, Stride 2	$20 \times \frac{T}{2} \times 64$
ResNetBlock-3	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 6$, Stride 2	$10 \times \frac{T}{4} \times 128$
ResNetBlock-4	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 3$, Stride 2	$5 \times \frac{T}{8} \times 256$
StatsPooling	–	10×256
Flatten	–	2560
Dense	–	256
Projection	–	N

初赛方案

训练阶段

- 数据准备
  - 90%train+10%dev
  - 每段音频，以3s为窗长进行滑动，并随机向前/后偏移0-1.5s，确保数据多样性
  - 随机施加噪声/混响增强
  - 在线数据增强，避免占用磁盘空间
- 模型设置
  - 80维fbank特征->预训练hubert large特征提取器1024维，使用fc层降维至80后再送入resnet backbone
  - Resnet34+TSTP生成256维方言embedding
  - 送入9分类器
  - 尝试embedding余弦距离作为训练损失，但效果不如分类损失，可能原因embedding维度远高于方言数量。若embedding的256维空间用于表示方言(9维)坐标，存在信息冗余

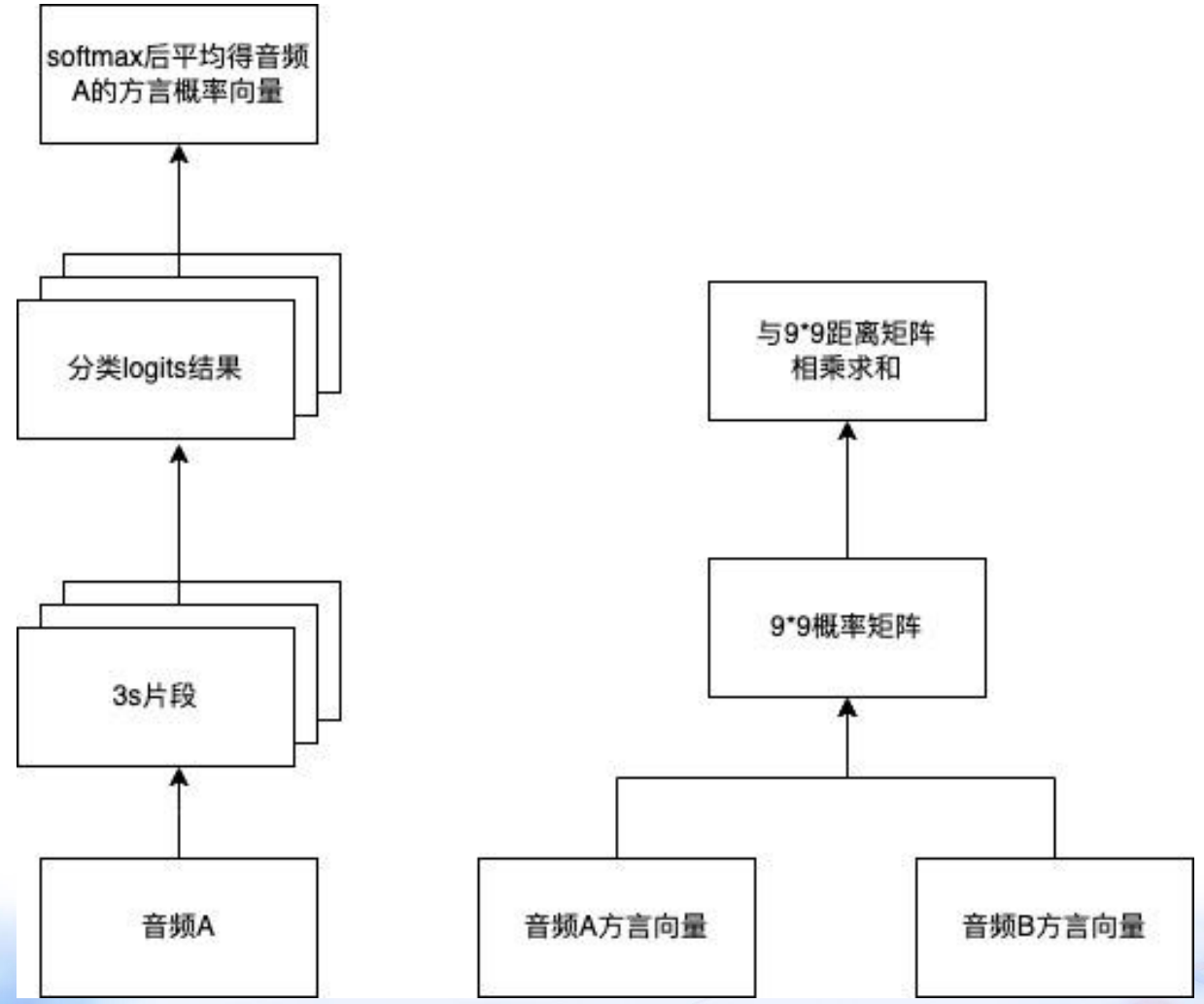


初赛方案

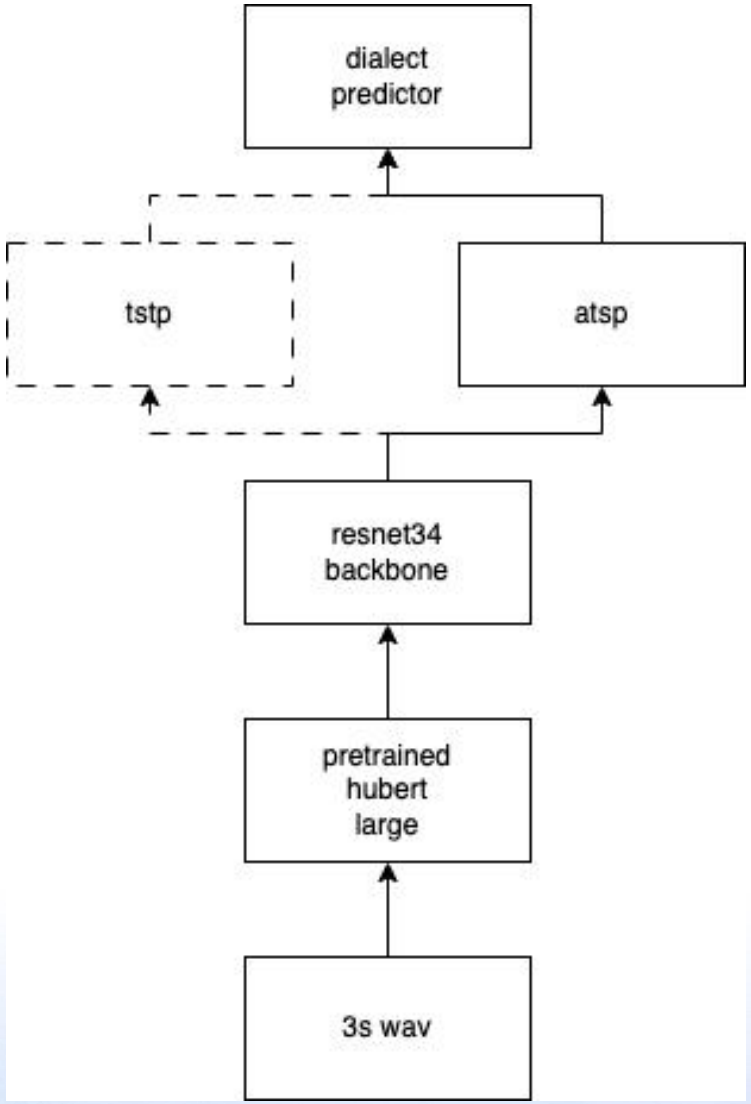
解码方式

- 选择dev loss最低的5个模型进行参数平均
- 以3s窗长，1.5s窗移，滑动提取每段音频的logits后验概率
- 若存在多段窗，logits做平均后生成softmax概率
- 两两音频之间，softmax概率做矩阵乘法，得9*9概率矩阵
- 概率矩阵与先验方言距离矩阵作点乘，并求和

初赛结果 MAE 7.31(rank5)



复赛改进



方案	MAE
初赛baseline	13.83
+ atsp	13.14
+ kespeech数据	12.82

改进(成功部分)

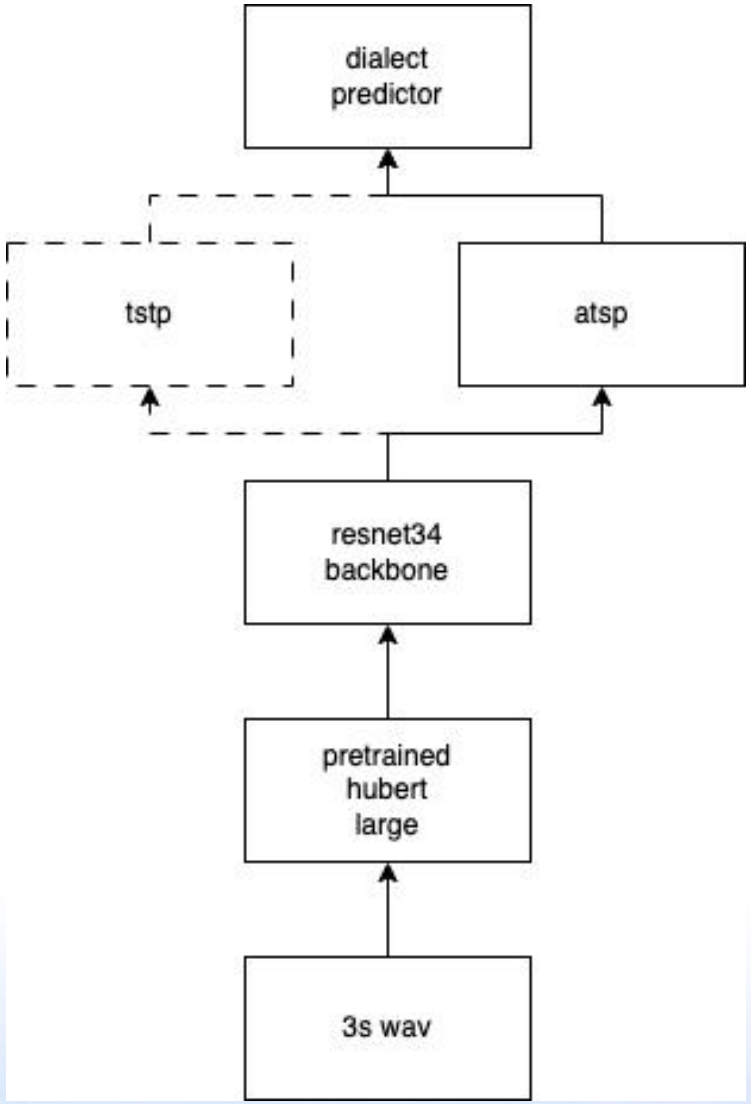
- astp替换为atsp

```
alpha = torch.tanh(self.linear1(x_in))
alpha = torch.softmax(self.linear2(alpha), dim=2)
mean = torch.sum(alpha * x, dim=2)
var = torch.sum(alpha * (x**2), dim=2) - mean**2
std = torch.sqrt(var.clamp(min=1e-7))
return torch.cat([mean, std], dim=1)
```

- 尝试mhatsp，但效果下降

```
for i, layer in enumerate(self.heads_att_trans):
    att_score = layer(chunks[i])
    alpha = F.softmax(att_score, dim=-1)
    mean = torch.sum(alpha * chunks[i], dim=2)
    var = torch.sum(alpha * chunks[i]**2, dim=2) - mean**2
    std = torch.sqrt(var.clamp(min=1e-7))
    chunks_out.append(torch.cat((mean, std), dim=1))
out = torch.cat(chunks_out, dim=1)
return out
```

复赛改进



方案	MAE
初赛baseline	13.83
+ atsp	13.14
+ kespeech数据	12.82

改进(成功部分)

- Kespeech标签体系
  - 8大方言体系+普通话->34个城市
- 根据比赛数据分布，增加kespeech数据
  - 普通话的10%数据
  - Kespeech标签与比赛标签重叠部分(9个)的全部数据
  - Kespeech中存在标签为方言，但无口音的情况，带噪训练
- Kespeech数据为16k采样率，为确保与比赛数据频谱一致，需降采样为8k后，再提取特征

复赛结果 MAE 12.82(rank8)

复赛改进

方案	测试集MAE	验证集loss	验证集MAE
复赛最佳方案	12.82	0.64	5.81
+ 0.9/1/1.1速度扰动	12.98	0.58	5.52
+ finetune hubert	13.19	0.53	4.9
+ kespeech multi task	13.33	0.54	5.1

改进(失败部分)

- 参考声纹速度扰动增强训练，验证集loss下降，测试集MAE上升
  - 速度扰动0.9/1/1.1，标签不变
- 尝试参考声纹中速度扰动扩充标签方式训练，效果不佳，验证集loss上升
  - 速度扰动0.9/1/1.1，标签扩充为3倍

复赛改进

方案	测试集MAE	验证集loss	验证集MAE
复赛最佳方案	12.82	0.64	5.81
+ 0.9/1/1.1速度扰动	12.98	0.58	5.52
+ finetune hubert	13.19	0.53	4.9
+ kespeech multi task	13.33	0.54	5.1

改进(失败部分)

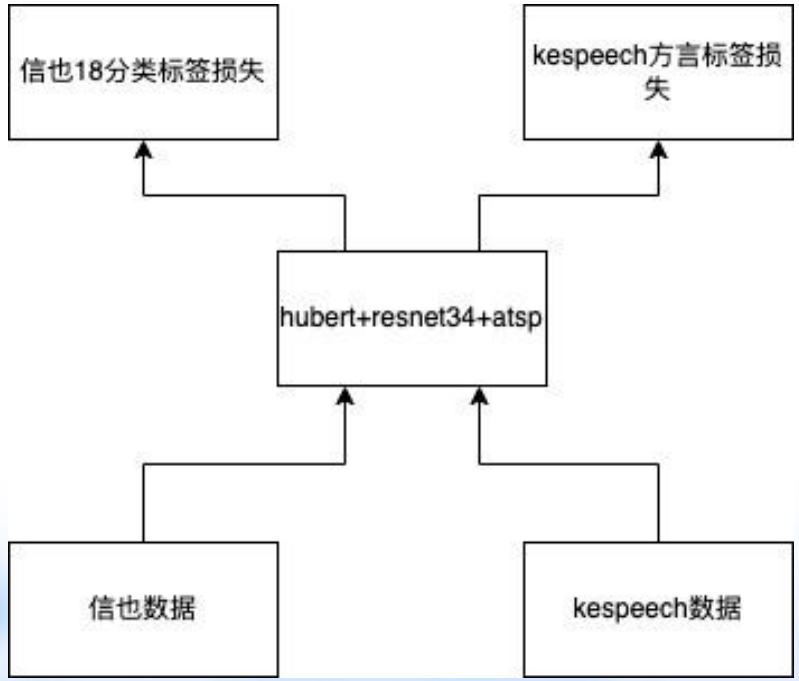
- 以1e-5学习率finetune hubert参数，验证集loss下降，测试集MAE上升
  - Hubert model使用layer norm，训练效果对batch_size不敏感
  - Resnet使用batch norm，训练效果对batch size敏感
  - 由于显存限制，hubert finetune时，resnet部分freeze，避免batch size太小导致收敛问题
- Hubert参数固定时，需设置为eval模式，确保特征提取的一致性

复赛改进

方案	测试集MAE	验证集loss	验证集MAE
复赛最佳方案	12.82	0.64	5.81
+ 0.9/1/1.1速度扰动	12.98	0.58	5.52
+ finetune hubert	13.19	0.53	4.9
+ kespeech multi task	13.33	0.54	5.1

改进(失败部分)

- 使用kespeech普通话的10%数据+方言的全部数据，新增kespeech标签预测任务，作multi task训练，验证集loss下降，测试集MAE上升



复赛改进

方案	测试集MAE	验证集loss	验证集MAE
复赛最佳方案	12.82	0.64	5.81
+ 0.9/1/1.1速度扰动	12.98	0.58	5.52
+ finetune hubert	13.19	0.53	4.9
+ kespeech multi task	13.33	0.54	5.1

改进(失败部分)

- 怀疑train/dev随机区分导致说话人泄漏
  - 原原初赛测试集中的新增9个方言数据加入训练
  - 原初赛测试集中的9个原有方言数据作为验证，该部分数据与训练数据不存在说话人重叠
  - 仍然验证集loss下降，测试集MAE上升

总结

可借鉴部分

- Hubert预训练特征提升较大
- 数据增强(噪声/混响)，以及随机施加0.9/1.1速度扰动，可以进一步降低验证集loss
- 引入kespeech数据，无论是作为带噪数据直接加入训练，或是新增分类任务作多任务训练，可以进一步降低验证集loss
- Hubert finetune也会进一步降低验证集loss

教训

- 验证集指标与系统测试结果相悖，系统测试次数有限且耗时较长，导致无法得出合理的迭代方向
- 初步怀疑测试集属于OOD问题，或验证集存在说话人泄漏导致loss偏低，可能需要更合理的迭代loss指标

信也科技杯 FINVOLUTION BIG DATA COMPETITION | 信也 FINVOLUTION

THANK YOU