



电子科技大学

University of Electronic Science and Technology of China

计算机学院 • 人工智能应用与挑战课程

Northeastern SMILE Lab - Recognizing Faces in the Wild

项目中期报告

组名: Honkai StarRail

作者: 刘 *¹ 赵 **² 武 **³

指导老师: 文泉

¹电子科技大学计算机科学与工程学院数字媒体技术专业, 学号:2022**

²电子科技大学计算机科学与工程学院数字媒体技术专业, 学号:2022**

³电子科技大学计算机科学与工程学院计算机科学与技术专业, 学号:2022**

目录

1	前言	2
2	题目概况	2
	2.1 竞赛概况	2
	2.2 竞赛内容	2
	2.3 竞赛意义	3
	2.4 数据规模	3
	2.4.1 数据集	3
	2.4.2 提交格式	5
	2.5 现有结果	6
3	中期进展情况	7
	3.1 当前排名与结果	7
	3.2 解题模型设计	7
	3.2.1 模型介绍	8
	3.2.2 具体实现	9
	3.3 中期进展结果分析	11
	3.4 中期进展主要问题分析	11
4	中期进展后拟改进措施	12
	4.1 特征处理方式	12
	4.2 训练集错误处理	12
	4.3 重新选择模型	12
	4.4 组合模型	13
5	参考文献	13

1 前言

本篇报告为计算机学院 • 人工智能应用与挑战课程，Honkai StarRail 小组的项目中期报告，小组选题为“Northeastern SMILE Lab - Recognizing Faces in the Wild”，（详见网址：[小组选题](#)）。

本篇报告包括项目概况介绍、项目中期进展情况、中期进展后拟改进措施三部分。

2 题目概况

2.1 竞赛概况

我们的选题名为“Northeastern SMILE Lab - Recognizing Faces in the Wild”：基于人脸识别的血缘关系鉴定模型（图：[\(1\)](#)）。本次竞赛的主办方为东北大学的微笑实验室⁴，该实验室的研究课题有“Effects of Speech Modification Strategies on Vowel Acoustics for Adolescents with Cerebral Palsy”、“Development of Auditory-Perceptual Speech Features in Preschoolers with Typical Development”、“Innovations in the Evaluation and Management of Pediatric Dysarthria”等⁵。

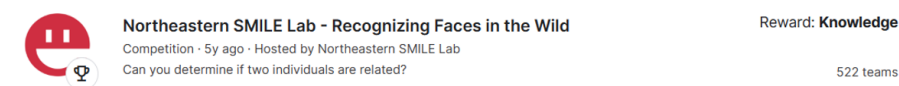


图 1: 竞赛题目

该竞赛的开始时间为 2019 年 5 月 14 日，结束时间为 2019 年 8 月 9 日，竞赛类型为 playground。虽然本次竞赛没有实物奖励，但由于该竞赛的题目较有意思，吸引了 522 支队伍、573 位选手参加该竞赛。

2.2 竞赛内容

在该竞赛中，题目要求参赛者训练一个模型图像识别模型，该模型能仅通过图像鉴定两人是否有血缘关系。由于用于亲属识别任务的现有图像数

⁴Northeastern SMILE Lab

⁵详见东北大学微笑实验室网址：<https://smilelab.sites.northeastern.edu>

据库规模不足以捕捉和反映世界各地家庭的真实数据分布，并且许多隐藏因素会影响家族面部关系，因此需要比计算机视觉算法更有区分度的模型，这也是本竞赛被提出的原因。

由于该竞赛考察的是参赛者模型的识别正确率，故该竞赛的评分标准为

$$Score = \frac{N}{M}$$

，其中 N 为正确识别血缘关系的测试集数量， M 为测试集总数。排行榜也是依据参赛者模型的识别正确率进行排名。

2.3 竞赛意义

当今社会，血缘关系鉴定有着多方面、多维度的社会意义：

- 当家庭中存在怀疑亲子关系的情况时，血缘关系鉴定有助于维护家庭的稳定与和谐。
- 当在法律上出现亲子权纠纷、抚养权争议等问题时，血缘关系鉴定有利于保障当事人的合法权益。
- 当对某些人的身份存疑时，如失踪人员寻亲、被拐卖儿童寻亲等，血缘关系鉴定可帮助确认个人身份与亲属关系。
- 在医学上可以帮助研究家族遗传病的遗传模式、基因突变的传播等，辅助控制家族遗传病的产生。

而现如今最常用的血缘鉴定方式为依据遗传学的基本原理，采用现代化的 DNA 分型检测技术来综合评判争议个体之间是否存在亲生、隔代或其他血缘关系⁶。这种血缘关系鉴定需提取被鉴定者的 DNA 样本，并通过 PCR 扩增、后 PCR 反应、毛细管测序仪检测等步骤完成血缘鉴定，鉴定过程较为繁琐。若能开发出一套基于被鉴定者的外貌特征而实现的亲子鉴定系统，则将大大提高血缘鉴定的效率，减少血缘鉴定的成本。

2.4 数据规模

2.4.1 数据集

官方给出的测试数据共包含 12 个文件（详见图 (2)），

⁶详见 <https://baike.120ask.com/art/50356>

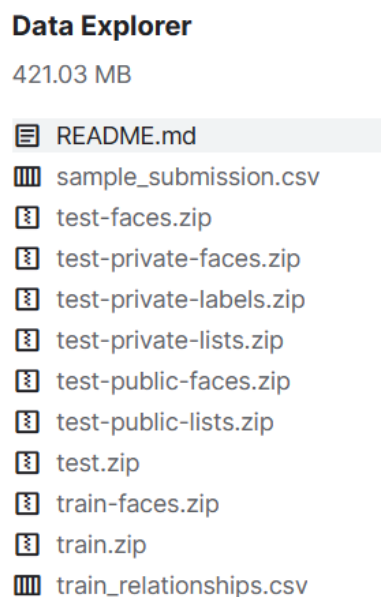


图 2: 测试文件

其中较为重要的分别是：

- train-faces.zip，训练集被分为家庭（F0123），然后是个体（MIDx）。同一 MIDx 文件夹中的图像属于同一个人。同一个 F0123 文件夹中的图像属于同一个家庭。
- train.csv，培训标签。其中需要注意的是，一个家庭中并不是每个成员都有亲属关系。例如，父母与他们的孩子有亲属关系，但彼此之间没有亲属关系。
- test-faces.zip，该测试集中包含未知个体的人脸图像。
- sample_submission.csv，该文件为一个正确格式的示例提交文件。列 `img_pair` 描述了图像对，即 `abcd-efgh` 表示图像对 `abcd.jpg` 和 `efgh.jpg`。我们的目标是预测 `test-faces` 中每对图像是否相关，其中 1 表示相关，0 表示不相关。

以上数据文件总大小为 421.03MB。

2.4.2 提交格式

在该竞赛中，要求的提交格式如图 (3)：

```
img_pair,is_related
X3Nk6Hfe5x-qcZrTXsfde,0.0
X3Nk6Hfe5x-LD0pWDM8w_,0.0
X3Nk6Hfe5x-PHwuDtHyGp,0.0
X3Nk6Hfe5x-L061N_U4ot,0.0
etc.
```

图 3: 提交格式

其中 `img_pair` 为测试集中的图像对，图像对中有需要判断是否存在血缘关系的两张图片，`is_related` 则表示该图像对中的两人是否存在血缘关系，是为 1，否为 0。在官方给出的提交样例文件 `sample_submission.csv` 中，我们可以看到 `img_pair` 与 `is_related` 两个列向量的具体值，如图 (4)

	A	B
1	img_pair	is_related
2	face05508	0
3	face05750	0
4	face05820	0
5	face02104	0
6	face02428	0
7	face01219	0
8	face04262	0
9	face03697	0
10	face03524	0

图 4: 提交样例

但在我们实际的提交过程中发现，列向量 `is_related` 的数值可以不为 0

或 1，若其值为 `img_pair` 图像对具有血缘关系的可能性，这一文件也是可以被正确提交的，如图 (5)

	A	B
1	<code>img_pair</code>	<code>is_related</code>
2	<code>face05508</code>	0.006345
3	<code>face05750</code>	0.722867
4	<code>face05820</code>	0.536038
5	<code>face02104</code>	0.838642
6	<code>face02428</code>	0.461341
7	<code>face01219</code>	0.025343
8	<code>face04262</code>	0.282194
9	<code>face03697</code>	0.068675
10	<code>face03524</code>	0.66064

图 5: 实际提交文件

2.5 现有结果









#	△	Team	Members	Score	Entries	Last Solution
1	▲ 2	mattemilio		0.923	169	5y 
2	—	lynesyChen		0.920	173	5y
3	▲ 2	maki		0.919	40	5y
4	—	AlexeyK		0.918	64	5y 
5	▲ 10	pi-null-mezon		0.917	111	5y 

图 6: 排行榜 1-5 名

截至目前，共有 573 位参赛选手，522 支参赛队伍。由于该题在五年前已完赛，故参赛选手的最终提交时间均在五年之前。

在该题目的 leaderboard 中（见图 (6)），目前的世界第一为 mattemilio，

其得分为 0.923 分，前五名得分均在 0.917 之上，前一百名得分均在 0.894 之上。他们的模型识别正确率均能达到九成左右。

3 中期进展情况

3.1 当前排名与结果

本小组中期时采用的初始模型的 Private Score 为 0.868，在 Private 测试集上排名第 189，Public Score 为 0.862，在 Public 测试集上排名第 171。模型和数据处理方法尚未找到最优解，项目尚未成熟，因此本小组的项目仍有很大的提升空间。在中期之后，我们会考虑采用更优的数据处理方法，对模型进行不断优化。






Submission and Description		Private Score	Public Score	Selected	
 submission.csv Complete (after deadline) · 1h ago		0.868	0.862	<input type="checkbox"/>	
188	- 38 samsudhin 		0.869	4	5y
189	- 25 Chinmay Vadgama 		0.868	1	5y
170	Captain Harlock 		0.863	4	5y
171	Or Com 		0.862	7	5y

图 7: 中期排名与结果

3.2 解题模型设计

由于本项目是基于人脸识别技术的研究项目。我们选择了基于 VGG-face 数据集预训练的 ResNet50 模型作为基础模型，并借助 Keras 深度学习框架构建了孪生神经网络结构。在模型构建过程中，我们去除了原始模型的输出层，保留了特征提取层，并设计了自定义的输出层模型，最终采用 sigmoid 层作为输出。这一设计旨在通过迁移学习，充分利用已有模型在大规模数据集上学习到的特征表达能力，从而提高模型的判断准确度。我们的目标是通过人脸图像进行血缘关系的判断，因此模型的输出层需要能够有效地区分不同人脸图像之间的相似度。采用孪生神经网络结构，可以使模型在学习过程中更好地捕捉到人脸图像之间的相似性特征，从而实现更精准的血缘关系判断。

3.2.1 模型介绍

1. ResNet50 模型

ResNet50 是 ResNet 系列中的一个特例，它有 50 层深，并且采用了残差模块的结构，克服了深度神经网络训练过程中的梯度消失和梯度爆炸等问题。通过使用残差连接，ResNet50 模型使得神经网络可以更深更广地构建，从而在图像识别等任务中取得了更好的性能。

ResNet50 模型的结构相对简单，主要由一系列的残差模块组成，其中包括了卷积层、批量归一化层和激活函数。整个网络的结构可以分为预处理层、卷积层、全局平均池化层和全连接层几个部分。具体结构见图 (8)

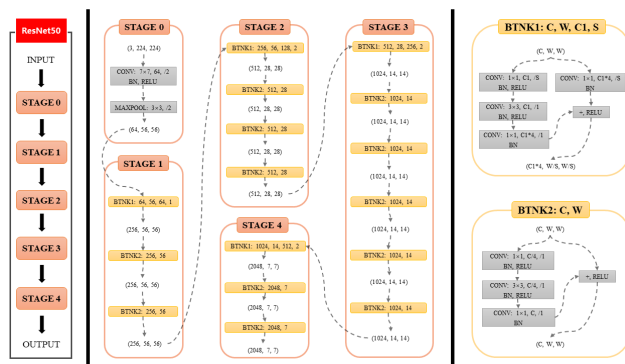


图 8: ResNet50 结构

2. 孪生神经网络

孪生神经网络 (Siamese Neural Network) 是一种特殊的神经网络架构，最初用于解决用于度量学习 (Metric Learning) 的问题。孪生神经网络由两个完全相同的子网络组成，它们共享相同的权重和参数。这两个子网络被称为“孪生网络”，因此得名“孪生神经网络”。

孪生神经网络有两个输入 (Input1 and Input2)，将两个输入 feed 进入两个神经网络 (Network1 and Network2)，这两个神经网络分别将输入映射到新的空间，形成输入在新的空间中的表示。通过 Loss 的计算，评价两个输入的相似度。(详见图 (9))

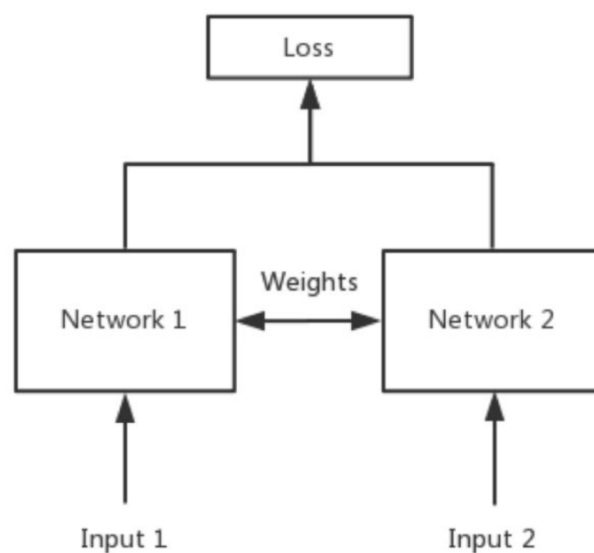


图 9: Siamese network

3.2.2 具体实现

首先，我们划分了训练数据集 `train_images` 和验证数据集 `val_images`，（见图 (10)）接下来我们定义一个用于读取图片的函数，该函数需采用已经

```

train_file_path = "../input/train_relationships.csv"
train_folders_path = "../input/train/"
val_families = "Fgg9"

all_images = glob(train_folders_path + "**/*.jpg")

train_images = [x for x in all_images if val_families not in x]
val_images = [x for x in all_images if val_families in x]

train_person_to_images_map = defaultdict(list)

ppl = [x.split("/")[-3] + "/" + x.split("/")[-2] for x in all_images]

for x in train_images:
    train_person_to_images_map[x.split("/")[-3] + "/" + x.split("/")[-2]].append(x)

val_person_to_images_map = defaultdict(list)

for x in val_images:
    val_person_to_images_map[x.split("/")[-3] + "/" + x.split("/")[-2]].append(x)

relationships = pd.read_csv(train_file_path)
relationships = list(zip(relationships.p1.values, relationships.p2.values))
relationships = [x for x in relationships if x[0] in ppl and x[1] in ppl]

train = [x for x in relationships if val_families not in x[0]]
val = [x for x in relationships if val_families in x[0]]
  
```

图 10: 划分训练数据集和验证数据集

封装好的 `preprocess_input` 函数。此外，我们还定义了一个生成器，这个生成器的作用是生成训练样本和测试样本。并且要设定相同的正负样本数。通过使用 `yield` 语句，能够在迭代的过程当中动态地生成数值。这种方式尤其适用于存在大量数据的情形，它能够助力节省内存空间。

下面载入源自 VGG-face 数据集预训练的 ResNet50 模型，进而对模型予以定义。其基本思路为运用若干个 `Concatante` 层，之所以如此，是因为并不知晓哪种池化方法更为适宜，也不明确池化后的特征相乘后做差的效果更好，还是做差后相乘的效果更优，故而将这几种处理方式径直进行连接，交由网络自行学习。并采用二元交叉熵充当损失函数。该步骤的具体代码见图 (11)

```
def baseline_model():
    input_1 = Input(shape=(224, 224, 3))
    input_2 = Input(shape=(224, 224, 3))

    for x in base_model.layers[:]:
        x.trainable = True

    x1 = base_model(input_1)
    x2 = base_model(input_2)

    x1 = Concatenate(axis=-1)([GlobalMaxPool2D()(x1), GlobalAvgPool2D()(x1)])
    x2 = Concatenate(axis=-1)([GlobalMaxPool2D()(x2), GlobalAvgPool2D()(x2)])

    x3 = Subtract()(x1, x2)
    x3 = Multiply()(x3, x3)

    x4 = Multiply()(x1, x1)
    x5 = Multiply()(x2, x2)
    x6 = Subtract()(x4, x5)

    x = Concatenate(axis=-1)([x6, x3])
    x = Dense(100, activation="relu")(x)
    x = Dropout(0.02)(x)
    out = Dense(1, activation="sigmoid")(x)
    model = Model([input_1, input_2], out)
    model.compile(loss="binary_crossentropy", metrics=['acc'], optimizer=Adam(0.0001))
    model.summary() # 输出网络结构日志
    return model
```

图 11: 模型定义

然后我们定义了回调函数，用于保存模型和辅助模型训练，以及进行学习率的设置，代码见图 (12)

最后实施训练调参操作。首先将 `mode` 设置为 ‘train’，每个 epoch 迭代 200 次，总计训练 100 个 epoch。待参数调好之后，再将 `mode` 设置为 ‘train_valid’，运用全部的数据集重新展开训练，每个 epoch 迭代 300 次，同样训练 100 个 epoch。

训练完成后，用该模型对测试集进行预测即可得到输出文件。

```
# 定义回调函数
file_path = "vgg_face.h5"

# 用于保存最优模型
checkpoint = ModelCheckpoint(file_path, monitor='val_acc', verbose=1, save_best_only=True, mode='max')

# 用于跳出训练循环，使模型保存最优模型
reduce_on_plateau = ReduceLROnPlateau(monitor='val_acc', mode='max', factor=0.1, patience=20, verbose=1)

def scheduler(epoch):
    # 每隔20个epoch，学习率减小为原来的1/10
    if epoch % 20 == 0 and epoch != 0:
        lr = K.get_value(model.optimizer.lr)
        K.set_value(model.optimizer.lr, lr * 0.1)
        print("lr changes to {}".format(lr * 0.1))
    return K.get_value(model.optimizer.lr)

# 用于在每隔指定epoch后自动减小学习率
learningratescheduler = LearningRateScheduler(scheduler)

callbacks_list = [checkpoint, reduce_on_plateau]
```

图 12: 回调函数

3.3 中期进展结果分析

在模型训练的过程中我们发现，当不知道如何处理特征更合适时，可以尝试将不同的特征处理方式串联在一起，交给网络自行去训练，这样可能会得到不错的准确率。

其次，通过使用多个 Concatenate 层，将不同的特征处理方式的结果直接连接在一起，让网络自行学习并权衡各种特征处理方式的重要性。这样的方法可以让模型在训练中学习到的更复杂且丰富的特征表示，提高模型的泛化能力。

最后，通过多个 Concatenate 层将不同的特征处理方式连接在一起，可以将不同的处理方式得到的特征融合在一起，这样模型就可以同时考虑多种特征处理方式所得到的特征表示。这种做法旨在让模型自行选择最合适的特征表示方式，从而更好地适应数据。

3.4 中期进展主要问题分析

在模型训练的过程中，我们遇到的主要问题是所采用的来自 GitHub 中的 ResNet50 模型所使用的库版本较为陈旧，致使在编译代码时遭遇诸多版本不兼容的问题。除此之外，或许存在效果优于 ResNet50 模型的其他选择，后续我们将会尝试采用其他模型来训练。

4 中期进展后拟改进措施

4.1 特征处理方式

在处理特征矩阵 x_1 与 x_2 时，我们仅采用了 $(x_1 - x_2)^2$ 和 $x_1^2 - x_2^2$ 的处理方法，但当我们浏览该比赛的讨论区时，我们发现有的参赛队伍对特征矩阵 x_1 与 x_2 进行了 $x_1 \times x_2$ 的操作，后续我们会尝试对特征矩阵进行 $x_1 \times x_2$ 的操作，观察模型的得分是否会变高。

4.2 训练集错误处理

我们浏览该比赛的讨论区时还发现，官方给出的原始数据中有一些明显错误，如图 (13)，图片中的两人明显不为同一人，但官方错误将其归类为同一人，这明显导致了我们的模型的错误训练。后续我们会尝试直接从官方给出的数据文件夹中删除那些错误的图片，重新训练模型。

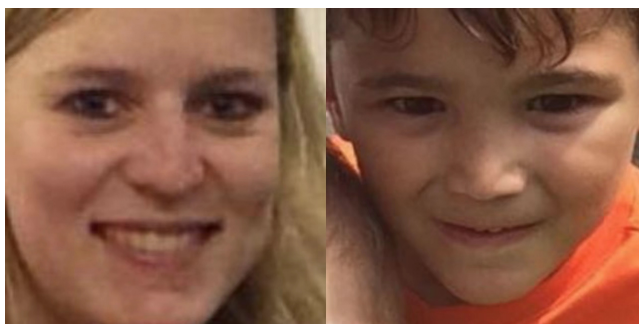


图 13: 错误样例 1

4.3 重新选择模型

经过我们小组成员的讨论，我们决定尝试将 ResNet50 (Residual Network) 模型替换为 SENet50 (Squeeze-and-Excitation Network) 模型。因为 SENet 引入了一种称为 “Squeeze-and-Excitation (SE)” 机制的注意力机制，这允许网络学习到每个通道的重要性。使得网络能够更加自适应地关注不同通道之间的关联性，增强特征的表达能力，进而提高网络性能。

4.4 组合模型

除此之外，也可以尝试将两个模型组合为一个新的模型。对于每个基于 VGGFace 的模型 x ，都有一个基于 Facenet 的类似模型 y 。对于这两种模型，得分是相似的，但两个结果几乎无关。作为进一步的步骤，VGGFace 和 Facenet 特征可以在同一模型中组合在一起以生成第三模型 z 。而第三模型 z 可能同时具有 x 与 y 的优点，这可能对我们提升成绩有所帮助。

5 参考文献

References

- [1] Koch, G., Zemel, R., & Salakhutdinov, R. (2015). Siamese Neural Networks for One-shot Image Recognition. In Proceedings of the 32nd International Conference on Machine Learning (ICML) (Vol. 37, pp. 1774-1782).
- [2] Song, F., Liu, W., Nie, F., Zhang, W., & Xu, Z. (2016). Online pair-wise deep hashing with exclusive cross-modal constraints. In Proceedings of the IEEE conference on computer vision and pattern recognition.
- [3] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition.
- [4] Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition.
- [5] Zagoruyko, S., & Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition.
- [6] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition.