# 项目背景

图像识别或图像分类是属于视觉科学和电脑科学领域的一个问题。这类问题的目的是需要将图片分类至一个或多个类别当中。这个问题可以通过人工手动或借助算法来完成，这些图片可能包含了不同的物体，图片的大小可能不一致，颜色上也千差万别。图片仍可以根据它们所包含的边缘（edges），角（corners），区域（blobs）和脊（ridge）来做分类[1]。

本项目将会采用广泛使用的机器学习算来进行图片识别以及图片分类。机器学习包含许多的分类算法比如，决策树，KNN，朴素贝叶斯和神经网络等等，每个模型都有它们各自的优缺点。这里我们将采用卷积神经网络（简称 CNN 或 ConvNet）来做图片分类。卷积神经网络是一个兼具深度和高识别率的神经网络，已经成功应用于图像识别领域。

本项目我们将基于 Kaggle 的一个竞赛项目 Dogs vs Cats[2] 来进行图像识别。这个项目包含了 25000 张的训练集图片以及 12500 张的测试集图片。这些图片将被划分到猫或狗的一个分类中。

# 问题描述

猫狗识别的项目属于监督学习中的分类问题，本项目中一共有 2 个类别，每一张图片只属于其中一个类别，我们的目标是利用 CNN 将图片划分到它们正确的类别中。因为从零开始训练一个卷积网络需要花费大量的时间和资源，这里我们会使用已经身经百战的 CNN 模型，并利用这些模型导出我们数据集的深度特征。然后基于这些深度特征来训练我们自定义的模型，并在最后验证我们的模型。

本项目会使用谷歌的 TensorFlow[3]作为神经网络的平台，使用 Keras[4]作为我们的 API 来构建并训练 CNN。这里我们将使用 VGG[5]，ResNet50[6]，InceptionV3[7]，Xception[8]作为我们的基础模型，这些模型已经全部封装在 Keras 的 Application API[9]中了。

# 输入数据

项目的数据集能直接从 Kaggle[10]上下载。Kaggle 一共提供了 3 个文件，train.zip 是训练集，test.zip 是测试集以及一个 csv 格式的 submission 文件。训练集包含了 2 个类别的图片，猫和狗的图片各 12500 张。测试集包含了 12500 图片，但没有提供任何分类信息。

这里我们将使用 Keras 的图片预处理 API ImageDataGenerator[11] 中的方法 flow_from_directory 加载我们的数据集。首先需要将训练集划分到 2 个子文件夹中，猫的图片放入 cat 的文件夹中，狗的图片放入 dog 文件夹中，测试集全部划分到 1 个子文件夹中。同时我们可以利用 flow_from_directory 的参数 target size 来调整我们所需要图片的大

小，因为不同的 CNN 需要的输入的图片大小会有所区别，比如 VGG 和 ResNet 要求的图片大小为（224,224）而 Xception 和 Inception 需要的图片大小为（299,299）。在利用 flow_from_directory 读取我们的图片后，我们调用 model 的方法 predict_generator 来进行预测。因为我们的模型是没有包含 top layer 的，所以得到会是所有训练集基于当前模型的一个深度特征。为了方便调试，我们可以把这些深度特征作为数组以文件的方式保存在本地。

基于这些深度特征，我们再来调试自己的模型，这里我们不需手动将训练集划分为训练数据和验证数据，Keras 的 Model API[12]提供一个训练模型的方法 fit。而 fit 中提供了一个参数 validation_split 可以自动帮助我们来划分训练数据和验证数据。比如 validation_split = 0.2 时，代表 80%的数据用于训练，而剩下 20%数据用于验证。

# 解决办法

因为我们基于 VGG，ResNet，Xception，Inception 导出的深度特征来构建模型。这使得我们的模型将比较简单，只需要构建最后一层，全连接层来做分类就行了。这里有 2 个方案，第一个方案是，我们只利用一个模型导出的深度特征来构建我们的模型。第二个方案是，我们利用多个模型导出的深度特征来构建我们的模型，甚至可以和其他机器算法集成。根据 Kaggle 竞赛者的采访[13]，第二种方案将会比第一种略微好一些。这里我们将会同时尝试 2 中方案。

# 基准模型

这里项目的要求是达到 Kaggle 的排行榜的前 10%，所以本项目的要求是在测试集上的 LogLoss 表现要低于 0.06127。

# 评估标准

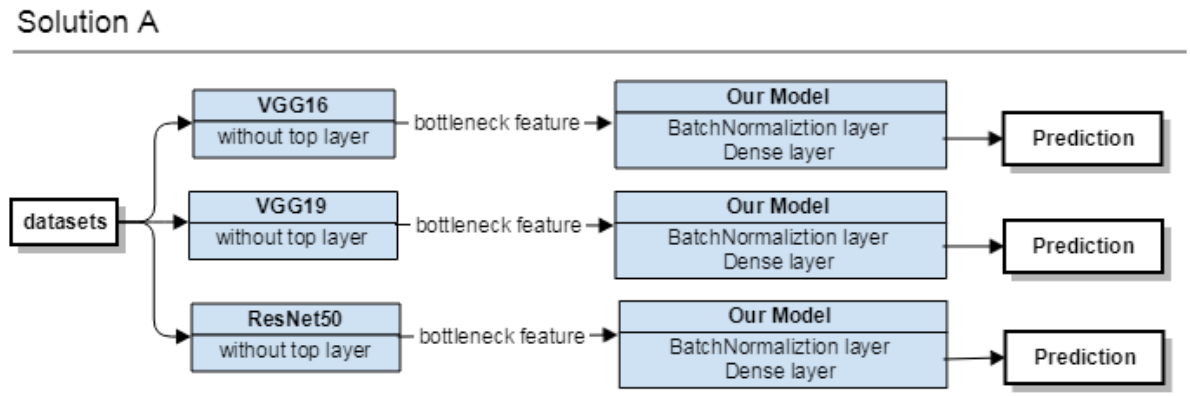
这里 Kaggle 官方给出的评估标准是 Log Loss 而且同时我们的测试集是没有标签的。所以我们也使用 LogLoss 作为我们的评估标准，LogLoss 的公式如下，其中 n 代表的是 sample 的数量，i 代表的是当前 sample 的索引，$y_i$代表的是模型预测的结果，$\hat{y}_i$代表的是真实的标签。

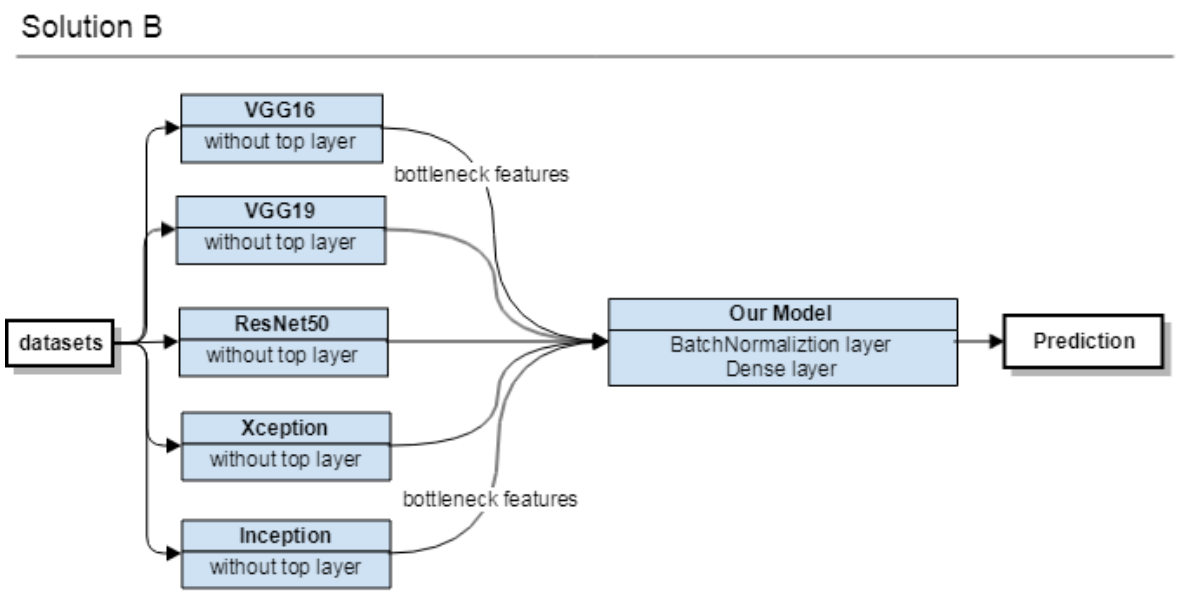
$$\text{LogLoss} = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)]$$

# 项目设计

项目设计如下图，方案 1 将使用单个模型导出的深度特征来做预测。方案 2 合并所有模型的导出的深度特征进行预测。我们的模型主要包含 2 层，第一层为 BatchNormailization

层，主要是为了防止过拟合。第二层为 dense 层，用于分类。当然在实际开发及调试过程中，下面的架构可能会略微调整。




# 引用

[1] Wiki page, Feature detection: en.wikipedia.org/wiki/Feature_detection_(computer_vision)

[2] Kaggle Project Dogs vs Cats: www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition

[3] TensorFlow official site: tensorflow.google.cn

[4] Keras official site: keras.io

[5] Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556, 2014

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep Residual Learning for Image Recognition. arXiv:1512.03385, 2015

[7] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. arXiv:1512.00567, 2015

[8] François Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. arXiv:1610.02357, 2016

[9] Keras applications introduction: keras.io/applications/

[10] Dogs vs Cats Datasets: www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data

[11] Keras ImageDataGenerator introduction: keras.io/preprocessing/image/

[12] Keras model API introduction: keras.io/models/model/

[13] Kaggle Team, Dogs vs. Cats Redux Playground Competition, Winner's Interview: Bojan Tunguz, 2017