CNet of Assiya Karatay

Term project paper

Assignment 2 9/3/19

Please use this template. Retain the gray text. Your new materials—in black 12-point Times New Roman—should not exceed 5 additional pages excluding references and figures. Note the evaluation criteria, and leave plenty of time for editing. There are two aspects to your term project: it should have function and it should be an opportunity to learn. You can refer to both in writing this up, as in the template.

Please paste your Assignment 1 here, including responses to each of your facilitator’s comments within each of their comments.

# ASSIGNMENT 2: PROJECT PROPOSAL PLUS

## 2.0 WHAT’S CHANGED

Provide no more than a page of 12-point type explaining what has been changed or added since assignment 1. Include in this whether and how the material in module 2 influenced this, or refer to reading that you did in working on this assignment (#2).

* Added Data Augmentation to train data
* Altered Callbacks:
  + Early stopping
  + Reduce learning rate on plateau monitoring val\_accuracy with minimum of 0,00001
  + Model checkpoint save at each epoch
* Dropout 25% after each conv layer
* Achieving subsampling with MaxPooling2D to reduce the noise
* Adding Batch Normalization after each convolutional layer increased test accuracy from 82% to 90%
* optimizer='adam'

## 2.1 SUMMARY DESCRIPTION, VERSION 2

One- or two-paragraph overall description of your proposed term project. Giving your application a name is usually a good idea.

CNet classification network predicts the classes of animals. We have two classes of dogs and cats. Given any image of pets, the application should be able to read it as data and predict the label.

The project is dedicated to design and analyze a model for binary classification. The data is from Kaggle competition dogs-vs-cats. It contains 25000 images labeled correctly.

The goal of the project is to solve the CAPTCHA challenge and demonstrate vulnerability of the system.

The objective is to design and analyze a model for binary classification

## 2.2 I/O EXAMPLES, VERSION 2

At least two specific examples of projected output for designated input. You will not be held to this—it is just explanatory at this point.

Изображение выглядит как кот, трава, собака, млекопитающее

Автоматически созданное описание

For example, given this input, the output for the first photo is cat, and for the second one it is a dog.

## 2.3 REQUIREMENTS, VERSION 2

High-level functional requirements statement in two roughly equal numbered lists, organized by triage. Separate your requirements into two approximately even categories (select modest “definite” requirements, otherwise “nice-to-do”). This organization allows you to first attain readily do-able goals without getting bogged down, and then move on to other goals if you can. State requirements in declarative language such as “(Recognize 0-9): The application will recognize numbers 0-9 from a 12 by 35 array of black-or-white pixels” (not “First I will build a neural net”). Giving each requirement a label (e.g., “(Recognize 0-9):”) helps with clarity and readability.

### 2.3.1 Definite Requirements (first priority)

(Recognize cat or dog): recognize cat and dog from 128\*128 array of RGB pixels. Digital image contains three dimensional array which is for width, height, and depth of color.

(Data split) We divided into three subsets because validation data is exposed to the model during training and best hyperparameters were chosen based on the highest accuracy on the val data. Then we evaluated our model on test data to know about overfitting, underfitting and get different metrics.

(Extract data) Get the data without saving it locally, so the program can be run anywhere with supported Python environment.

(Connect to GPU) The program checks if it is connected to GPU since the model gets terminated without enough memory and computation units.

(Define callbacks) Save the model after each epoch to ensure that we can continue from the point it stopped forcefully.

### 2.3.2 Nice-to-do (second priority)

(Augment data): Add horizontal and vertical shifts, horizontal flip to get better results of the model

(Evaluate the model) Confusion matrix, classification report including the accuracy, loss, f1 score, precision, and recall.

(Show wrong predicted images) Initially, predicted labels added to the data frame of true labels. Based on this data, visualize the data which is predicted wrong by the model.

## 2.4 V2: HOW SUCCESS WILL BE ASSESSED

Explain, as specifically as possible (quantification is ideal) how success of the project should be assessed. We don’t want aimless projects of the form “I’ll play around with X until time runs out” because they are less motivational and because you learn less.

Accuracy around 90% would be considered as success. We achieved it on test data which is not exposed during the training of the model.

## 2.5 V2 TECHNOLOGY EXPLANATION

Explain what two technologies you intend to use--and why you feel they apply to your particular project. One of the two may be emphasized as the implementation and the other as an alternative or as a complement—discussed but not implemented if need be. If possible, show fragments of code execution. For example, if you are using TensorFlow, show that you have run some code. This can be simple—we just want you to break the ice with implementation.

The current version of the code can be found in the following link:

<https://github.com/divassya/CS767_ML_project.git>

1. I use Tensorflow Keras API to build, compile, and evaluate the model [1].

model = tf.keras.models.Sequential([

    # Note the input shape is the desired size of the image with 3 bytes color

    # This is the first convolution

    tf.keras.layers.Conv2D(64, (3,3), activation='relu', input\_shape=(128, 128, 3)),

    tf.keras.layers.MaxPooling2D(2, 2),

    # The second convolution

    tf.keras.layers.Conv2D(64, (3,3), activation='relu'),

    tf.keras.layers.MaxPooling2D(2,2),

    # The third convolution

    tf.keras.layers.Conv2D(128, (3,3), activation='relu'),

    tf.keras.layers.MaxPooling2D(2,2),

    # The fourth convolution

    tf.keras.layers.Conv2D(128, (3,3), activation='relu'),

    tf.keras.layers.MaxPooling2D(2,2),

    # Flatten the results to feed into a DNN

    tf.keras.layers.Flatten(),

    tf.keras.layers.Dropout(0.5),

    # 512 neuron hidden layer

    tf.keras.layers.Dense(512, activation='relu'),

    tf.keras.layers.Dense(1, activation='sigmoid')

])

1. I will use Kaggle API to extract the data from the Kaggle database. It was successful so we do not download any files manually.

# initialise the API

kag = KaggleApi()

kag.authenticate()

# downloading the files

kag.competition\_download\_files(competition='dogs-vs-cats', path='./')

# unzip the files

with ZipFile('dogs-vs-cats.zip', 'r') as z:

    z.extractall()

with ZipFile('train.zip', 'r') as z:

    z.extractall()

1. I use google.colab.drive library in Python to save the output files on personal drive. This is useful because when the session is terminated, output files on colab environment are deleted. In this way, we can have our necessary files on our side.

from google.colab import drive

drive.mount('/content/gdrive')

1. I use train\_test\_split function from scikit-learn to divide the data into parts used for training, testing, and validation

train\_val\_data, test\_data = train\_test\_split(df,

test\_size = 0.2,

stratify = df["label"],

random\_state = seed)

1. Data augmentation tools from keras are used to improve the accuracy. Data augmentation artificially increases the size of the training set by generating many realistic variants of each training instance. This reduces overfitting, making this a regularization technique.[4]

training\_datagen = ImageDataGenerator(

      rescale = 1./255,

      rotation\_range=20,

      width\_shift\_range=0.2,

      height\_shift\_range=0.2,

      shear\_range=0.2,

      zoom\_range=0.2,

      horizontal\_flip=True,

      fill\_mode='nearest')

validation\_datagen = ImageDataGenerator(rescale=1. / 255)

train\_generator = training\_datagen.flow\_from\_dataframe(

    dataframe = train\_data,

    directory = train\_path,

    x\_col = "image\_name",

    y\_col = "label",

    target\_size=image\_size,

    batch\_size=batch\_size,

    class\_mode='binary',

    shuffle=True,

    seed=seed)

## 2.6 V2 DATA SOURCES

Explain whether or not your project requires data. If so, describe were you will obtain it. Be careful about this because you won’t have a project if it needs data and you have to spend too much time hunting and gathering it.

The project requires images of two categories. The data is obtained from Kaggle competition datasets. The data is from CAPTCHA challenge that addresses problems such as reduce email and blog spam and prevent brute-force attacks on web site passwords.

The Asirra data set is manually classified by people. [3]

## 2.8 REFERENCES FOR PROPOSAL V2

Fill in, and also cite each of the following (e.g., “[2]“) within the text. References can include specific places in the notes and textbook. You are free to include references from the prior assignment version.

[1] Yamashita, R., Nishio, M., Do, R.K.G. *et al.* Convolutional neural networks: an overview and application in radiology. *Insights Imaging* **9,**611–629 (2018). https://doi.org/10.1007/s13244-018-0639-9

[2] S. Albawi, T. A. Mohammed and S. Al-Zawi, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.

[3] *Dogs vs. cats*. Kaggle. (n.d.). Retrieved April 26, 2022, from <https://www.kaggle.com/competitions/dogs-vs-cats>

[4] Géron, A. (2020). Data Augmentation. In *Hands-on machine learning with scikit-learn, keras and tensorflow: Concepts, tools and techniques to build Intelligent Systems* (pp. 450–452). O'Reilly.

## 

## 2.7 Instructor’s Evaluation of Assignment 2



Relevance