UiA Fakultet for teknologi og realfag

IKT115 - INTRODUKSJON TIL KUNSTIG INTELLIGENS-TEKNOLOGI

Prosjektoppgave

Spring 2024

Gormery K. Wanjiru
May 3, 2024

Obligatorisk gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

1.	Vi erklærer herved at vår besvarelse er vårt eget arbeid, og at vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.	Ja
2.	**Vi erklærer videre at denne besvarelsen:** • Ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. • Ikke refererer til andres arbeid uten at det er oppgitt. • Ikke refererer til eget tidligere arbeid uten at det er oppgitt. • Har alle referansene oppgitt i litteraturlisten. • Ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse.	Ja
3.	Vi er kjent med at brudd på ovennevnte er å betrakte som fusk og kan medføre annullering av eksamen og utestengelse fra universiteter og høgskoler i Norge, jf. Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	Ja
4.	Vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert.	Ja
5.	Vi er kjent med at Universitetet i Agder vil behandle alle saker hvor det forligger mistanke om fusk etter høgskolens retningslinjer for behandling av saker om fusk.	Ja
6.	Vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider.	Ja
7.	Vi har i flertall blitt enige om at innsatsen innad i gruppen er merkbart forskjellig og ønsker dermed å vurderes individuelt. Ordinært vurderes alle deltakere i prosjektet samlet.	Ja

Publiseringsavtale

Fullmakt til elektronisk publisering av oppgaven Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).
Oppgaver som er unntatt offentlighet eller taushetsbelagt/konfidensiell vil ikke bli publisert.

Vi gir herved Universitetet i Agder en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:	Ja
Er oppgaven båndlagt (konfidensiell)?	Nei
Er oppgaven unntatt offentlighet?	Nei

Abstract

In this report we go into image recognition. You'll get to see why its important and more than that a step by step recipe on how it works.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Image recognition is big field within ai that has perhaps the most noticable results. While today LLM are what are most famous today, you could say right before openai released gpt3 and further back, image recognition was what carried the ai interest baggage. Most people before then, including myself probably got intreseted and decided to study AI because of image recognistion. perhaps they saw something done and fell i love or thay needed a physical world task done etc.

Chapter 2

Theory and Goals

The image recognition im going to make is animal recognition. It is a simple project but can truly showcase the abilities of model, the affects of the training and how good the results are.
This is because Animals are diverse. They come in all different shapes and sized and also similar shapes and sizes. The model would be good if it can differentiate different shape and size animals but even better is it can differentiate similar ones. That is the goal.
Some categories its going to me trained on are varied from pets, to safari animals, asian animals ets.
The model we used is called animal-data[1] and it came from kaggle.
The datasets contains realatively good quality images and for each image it has different orianta-tions, brightness and reflections of it. this makes made the data set even more valuable as ut could gave four different pictures from the same picture. here are some example images in the mode:

Figure 2.1: Cover image of the animal-data[1] dataset on kaggle

Chapter 3

Methods in Trainig and Testing The Model

3.1 Methodology

For this project as in most image classification it can be split up into:

- Data Preparation
- Model Training
- Model Testing and Evaluation

3.1.1 Data Preparation

The dataset for this project, consisting of images of various animal species, was from Kaggle. Unfortunately, this dataset did not come pre-split into training and testing subsets. This meant i had to do i myself. I could have just used it as is, and for testing data use some from google, but i decided this was faster than going around installing images one by one from google images.
The dataset was structured with separate folders for each category of animals, each category folder containing respective images. So i created a python script that redistributed these images into two new folders: one for training and another for testing. 90% of the images were allocated to the training set, and the remaining 10% were reserved for the test set. I decided not to do the normal rule of thumb when splitting data 70:30 because images of all these animals are easily accessible from google. Also the dataset was only 38 megabyte, so i decided the model having more images to train om would increase the accuracy.
Fortunately sklearn has a function that does this already so i used that.

```
import os
import shutil
from sklearn.model_selection import train_test_split

current_script_path = os.path.dirname(os.path.abspath(__file__))

data_dir = os.path.join(current_script_path, 'animal_data')
print(data_dir)

categories = os.listdir(data_dir)

train_dir = os.path.join(current_script_path, 'training')
test_dir = os.path.join(current_script_path, 'testing')

os.makedirs(train_dir, exist_ok=True)
os.makedirs(test_dir, exist_ok=True)

for category in categories:
    category_path = os.path.join(data_dir, category)
```

```
        images = os.listdir(category_path)

        train_cat_dir = os.path.join(train_dir, category)
        test_cat_dir = os.path.join(test_dir, category)
        os.makedirs(train_cat_dir, exist_ok=True)
        os.makedirs(test_cat_dir, exist_ok=True)

        train_imgs, test_imgs = train_test_split(images, test_size=0.1, ...
            random_state=42)

        for img in train_imgs:
            shutil.copy(os.path.join(category_path, img), ...
                os.path.join(train_cat_dir, img))
        for img in test_imgs:
            shutil.copy(os.path.join(category_path, img), ...
                os.path.join(test_cat_dir, img))
```

After spliting i had 2 folder in the same structure as the original but different sizes

- training - 32.5 MB
- testing - 3.82 MB

3.1.2 Model Training

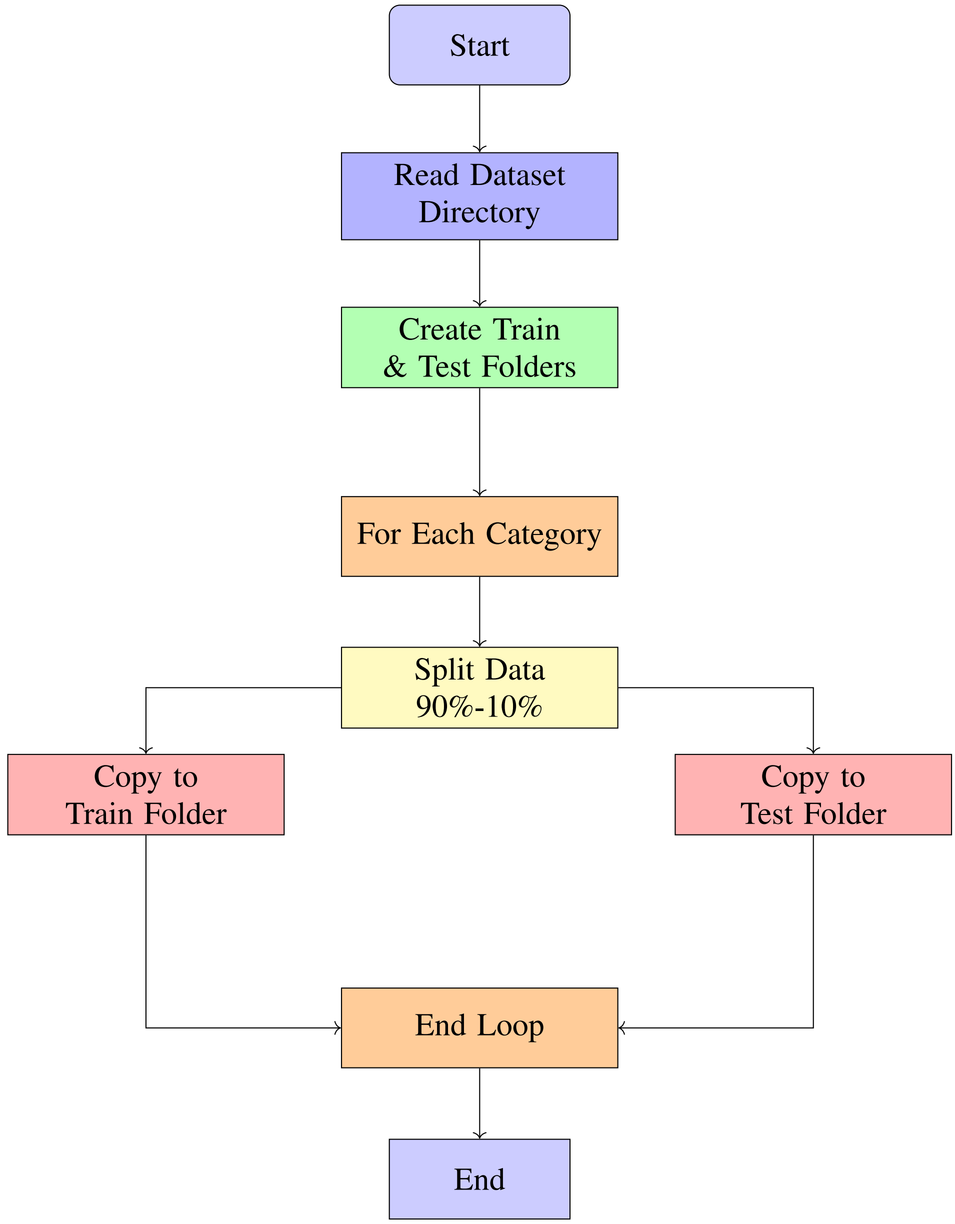
For training the model, I used Google's Teachable Machine. It's a nifty web-based tool designed for those who might not have deep coding knowledge but still want to play with creating their own machine learning models. It's pretty user-friendly and allows for quick image classification prototypes. I din't know about it before but now i love it.
Here's how it was done: The first time i uploaded the each category in the training dataset as a class one by one. luckily its not too big and took maybe a minute. The nex part was the training here you could go into the advanced setting but i tried first with base setting. Things like learning rate and the number of training epochs are all adjustable, letting you tweak the model just right
It even offers the under the hood which shows the statistics and more details.
In total the model was trained on 1743 images for 50 epochs using a batch size og 16 adn at a learning rate of 0.001

3.1.3 Model Testing and Evaluation

After the model was trained, it was time to put it to the test. I fetched the images that I'd set aside in the test set—these were the images the model hadn't seen before, to make sure I wasn't accidentally cheating by having it learn from its test papers.
The goal was to see how well this trained model could apply its 'knowledge' to new, unseen data. The tests involved calculating accurate and confident it was. In Teachable Machine they show it quite nicely as a bar.
This stage was crucial. Not only did it show how the model performs under new conditions, but it also highlighted areas where it could use a bit more training. Understanding the model's strengths and weaknesses helped me think about what tweaks might be needed to enhance its ability to recognize and classify new images accurately.

3.1.4 Data Preparation and Distribution Flow Diagram

the steps taken to split the dataset into training and testing sets:


Start
Read Dataset Directory
Create Train & Test Folders
For Each Category
Split Data 90%-10%
Copy to Train Folder
Copy to Test Folder
End Loop
End

Chapter 4

Results

4.0.1 Results and Discussion

So, I sent in at it at least 12 images for each category, just to see the basic stats like accuracy, and confidence. What do you know? It nailed the accuracy on the training set images—like a perfect 100%. It's good at what it learned, which was to be expected. But then, it was time to make this more interesting, let's make this a bit more interesting.
I tested it with some tricky images from Google, trying to trip it up a bit, and true? It stumbled sometimes. I used images of asses mostly but placed them in different settings to really push the limits. When the background was grassy or dry, the model thought the donkey was a kangaroo. When there was a noticeable hanging stomach, it guessed cow, and with mountainous or forest backgrounds, it went with deer. Interestingly, it never guessed horse, which would be the closest. So i investigated the training data for horses a bit. Turns out, in the training set, the horses were mostly either vibrant brown or white - and other vibrant colors, never the grey shades like those of a donkey. This little experiment showed the model does what it's supposed to—extract features and recognize them, but only within the context it was trained on.
When it comes tweaking things. I played around with different settings in the "Advanced" options—things like tweaking the number of training epochs. Increasing the epochs did bump up the accuracy initially, but then it plateaued, and going too long actually started to hurt the performance. This is the same with what I've learned about potential overfitting—too much learning isn't always a good thing.

What Worked and What Didn't

I have a few examples to show this off:

- **correct clasification:** There were cases where the model was spot on. I'll put those images right here in the report, and you can see how it got them right. it was the very different animals like birds, elephant etc.
- **Misclassifications:** Then there were the misfires. Like the donkey-dressed-as-a-deer kind of mistakes. I'll show these too, along with a discussion on why it probably messed up. the similar animals like deer, horse and shockingly kangaroo
- **Weird Stuff:** And for fun, I threw in some totally unrelated images just to see what it would do. The results were as mentioned before

Ethical and Practical Challenges

Discussing ethics, Here are three ethical challenges with this model:

1. **Bias in Training Data:**
2. **Misuse of Technology:**

3. **Data Privacy:** Where and how we source our images could should be thought about extensively.

this model has big potentions here are some just to mention afew:

1. More diverse data, especially with color variations.
2. Maybe tweaking the neural network architecture itself, adding some layers or playing around with the dropout rate.
3. also better curating of data. like in the case of hose i dint notice until the model was trained because the issue with the data what unnoticable normally

our model is decent at recognizing stuff, as long as it's stuff it has seen in somewhat similar contexts before. But throw something different, and it's a bit of a coin toss.

Chapter 5

Conclusions

when I added more images to the training dataset, the model's performance improved. More data, better learning, who would've thought, right?
generally a good learning experience

Appendix A

Use of ai:

It wasn't used

Bibliography

[1] Md. Khalequzzaman Sarker Likhon. *animal data*. URL: https://www.kaggle.com/datasets/likhon148/animal-data/data. (accessed: 03.05.2024).