Automated weather classification using transfer learning

IBMPROJECTREPORT

Submittedby

JEYA KRISHNAN S(Teamleader)	(961220104011)
S ARAVINTH	(961220104023)
MAGESWARAN S	(961220104015)
SAMUEL A	(961220104022)

Team Id number : NM2023TMID17790

LOYOLA INSTITUTE OF TECHNOLOGY & SCIENCE -THOVALAI KANNIYAKUMARI

ANNAUNIVERSITY,CHENNAI600025

JUNE– 2023

BONAFIDECERTIFICATE

Certified that this titled“ **Automated weather classification using transfer learning** ”is the bonafiderecord of words done by **JEYA KRISHNAN S,S ARAVINTH,MAGESWARAN S SAMUEL A** studying in third year sixth semester Computer science and engineering has completed **IBM PROJECT** during the academic year 2022- 2023 has been successfully.

Guide Mr.P.MURUGAN

ASSISTENT PROFESSOR

LOYOLA INSTITUTE OF TECHNOLOGY & SCIENCE.

ACKNOWLEDGEMENT

We thank the almighty, for the blessings that have been showered upon me to bring for the success of the project.

Wewouldliketoexpressmysinceregratitudetoourchairman and wouldliketothankandexpressourgratitudetoourprincipal forhis constant supporttomywork.

WewishtoexpressmythankstoourHeadoftheDepartment,,forherencouragementandsupporttocompletethisproject.

Weexpress our heartfelt gratitude to our guide, supervisor and internal guide Assistant Professor, Department of Computer Science & Engineering for priceless guidance and motivation which helped me to bring this project to aperfectshapewhoencouragedmeineachandeverystepofthisprojecttocompleteitsuccessfully.

ABSTRACT

Weather recognition is a common problem for many branches of industry. For example self-driving cars need to precisely evaluate weather in order to adjust their driving style. Modern agriculture is also based on the analysis of current meteorological conditions. One of the solutions may be a system detecting weather from image. Because any special sensors are needed, the system should be really cheap. Thanks to transfer learning it is possible to create image classification solutions using a small dataset. In this paper three weather recognition models are proposed. These models are based on InceptionV3, MobileNetV2 and ResNet50 architectures. Their efficiency is compared and described.

Keywords: Machine learning · Deep learning · Transfer learning · Image classification · Convolutional Neural Networks (CNN) · Neuralnetwork architecture · Weather classification

TABLEOFCONTENTS

1. INTRODUCTION

1.1 ProjectOverview:

Weather classification is an essential tool for meteorologists and weather forecasters to predict weather patterns and communicate them to the public. Weather phenomenon recognition notably affects many aspects of our daily lives, The analysis of weather phenomenon plays a crucial role in various applications, for example, environmental monitoring, weather forecasting, and the assessment of environmental quality. Besides, different weather phenomena have diverse effects on agriculture. Therefore, accurately distinguishing weather phenomena can improve agricultural planning.

1.2 Purpose:

- The user interacts with the UI to choose an image.
- The chosen image is processed by a VGG19 deep learning model.
- The VGG19 model is integrated with a Flask application.
- The VGG19 model analyzes the image and generates predictions.
- The predictions are displayed on the Flask UI for the user to see.
- This process enables users to input an image and receive accurate predictions quickly.

To accomplish this, we have to complete all the activities and tasks listed below

- Data Collection.
  - Download and extract the dataset.
- Image Pre-processing.
  - Import the required library
  - Configure ImageDataGenerator class
  - Apply ImageDataGenerator functionality to Trainset and Testset
- Model Building
  - Pre-trained CNN model as a Feature Extractor
  - Adding Dense Layer
  - Configure the Learning Process
  - Train the model
  - Check Model Accuracy
  - Save the Model
  - Test the model
- Application Building
  - Building HTML Pages
  - Building Flask Code
  - Run Application

.

2. IDEATION&PROPOSED SOLUTION

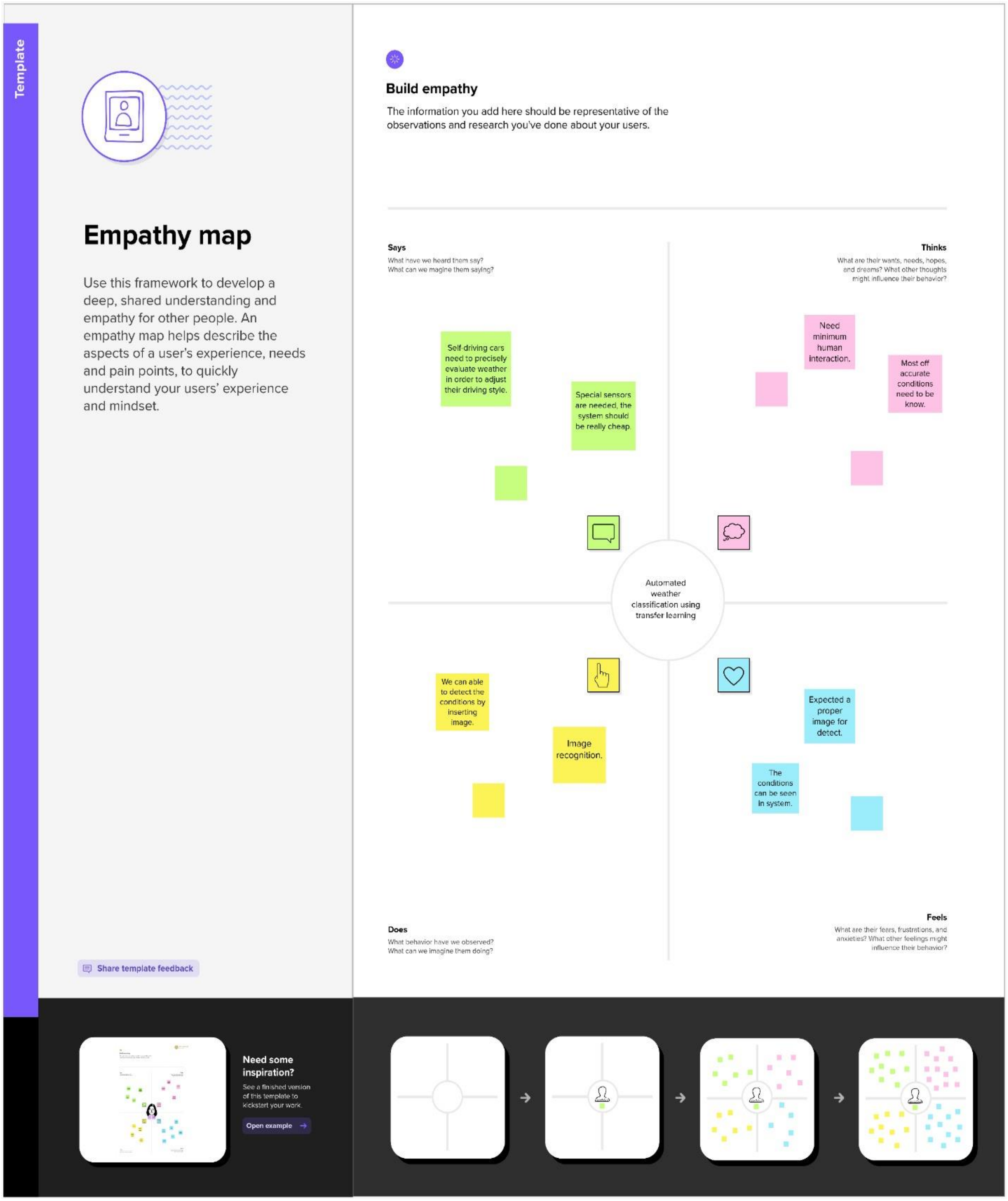
2.1 ProblemStatementDefinition:

Weather recognition is a common problem for many branches of industry. Modern agriculture is also based on the analysis of current meteorological conditions. One of the solutions may be a system detecting weather from image. Because any special sensors are needed, the system should be really cheap.

Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes me feel**
PS-1	I am self driver	I'm trying to detect the weather using transfer learning	But we need to learn about transfer learning	Because to detect weather using image	It is easy for self driving cars

PS-2	I am a normal person	Detect weather using image	We need a clear image	To detect weather	It’s almost easy for all

EmpathyMap:



Template

Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

Share template feedback

Need some inspiration?

See a finished version of this template to kickstart your work.

Open example

Build empathy

The information you add here should be representative of the observations and research you've done about your users.

Says
What have we heard them say?
What can we imagine them saying?

- Self-driving cars need to precisely evaluate weather in order to adjust their driving style.
- Special sensors are needed, the system should be really cheap.

Thinks
What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?

- Need minimum human interaction.
- Most off accurate conditions need to be know.

Automated weather classification using transfer learning

Does
What behavior have we observed?
What can we imagine them doing?

- We can able to detect the conditions by inserting image.
- Image recognition.

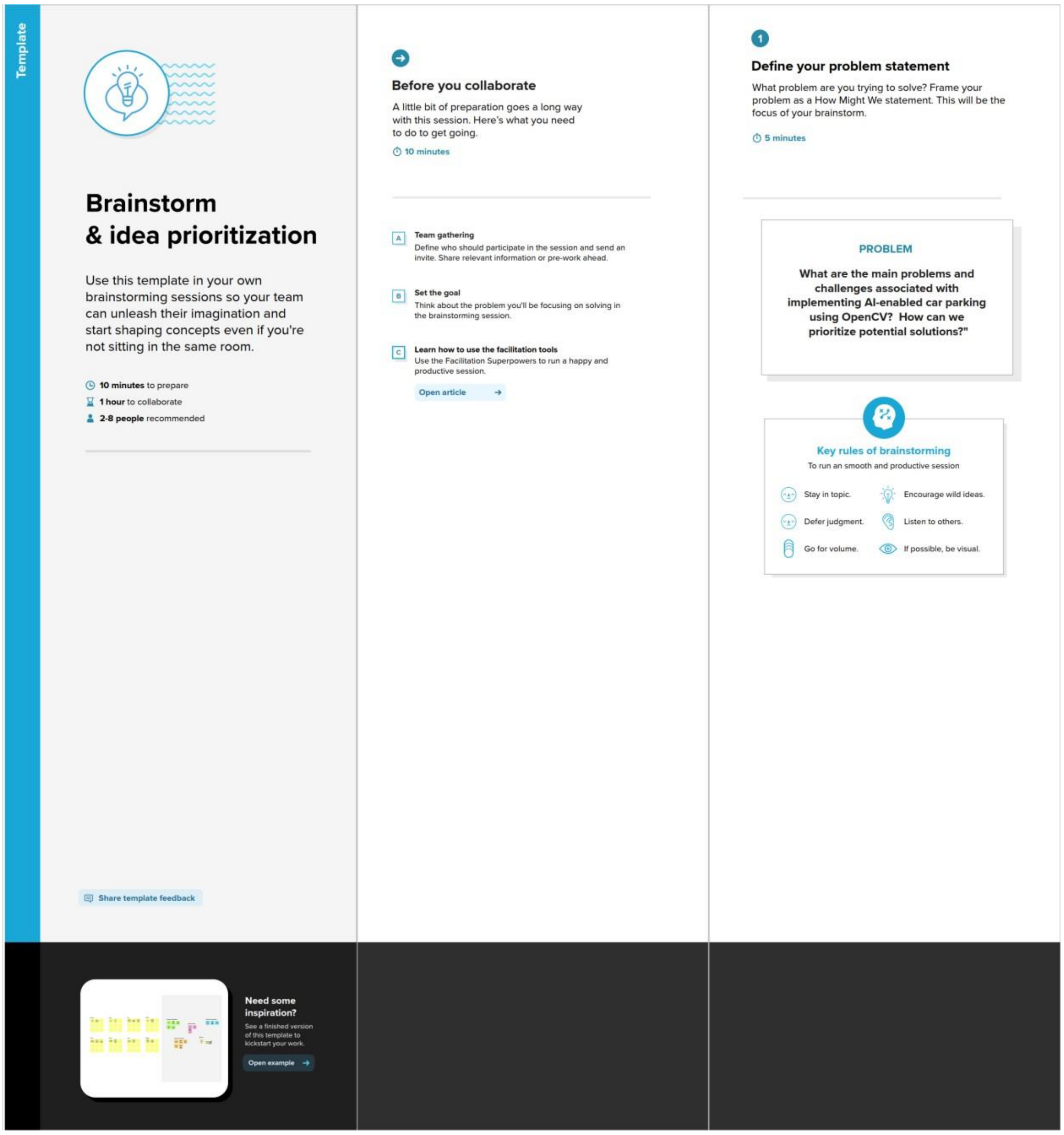
Feels
What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

- Expected a proper image for detect.
- The conditions can be seen in system.

2.2 Ideation&Brainstorming:

Brainstorm&IdeaPrioritization:

Step-1:TeamGathering,CollaborationandSelecttheProblemStatement



Template

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Share template feedback

Need some inspiration?
See a finished version of this template to kickstart your work.
Open example

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
10 minutes

A Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

C Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.
Open article

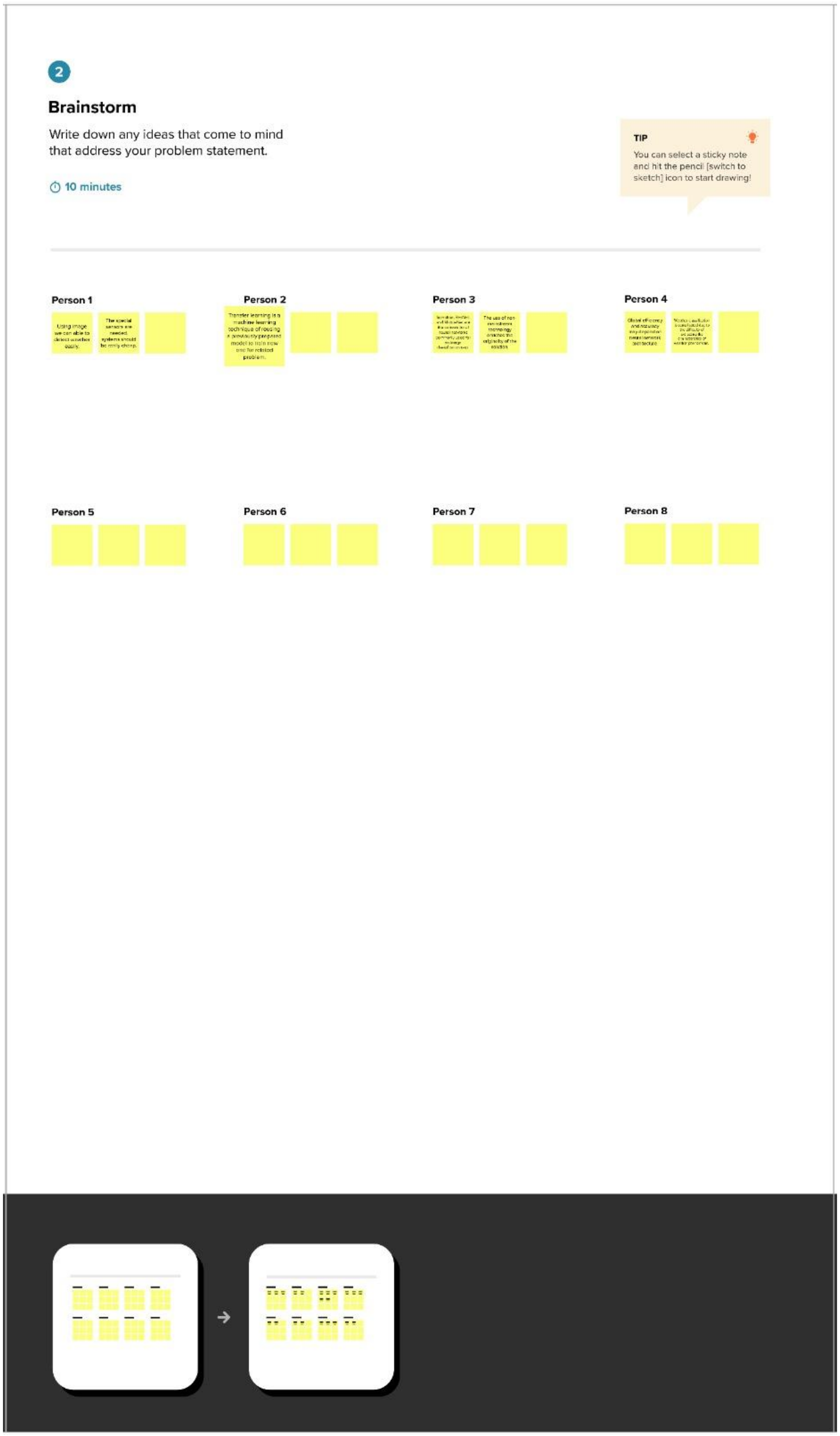
1 Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes

PROBLEM
What are the main problems and challenges associated with implementing AI-enabled car parking using OpenCV? How can we prioritize potential solutions?"

Key rules of brainstorming
To run an smooth and productive session

Stay in topic.	Encourage wild ideas.
Defer judgment.	Listen to others.
Go for volume.	If possible, be visual.

Step-2:Brainstorm,IdeaListingandGrouping



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Person 1

Using image we can able to detect weather easily.

The special sensors are needed. systems should be really cheap.

Person 2

Transfer learning is a machine learning technique of reusing a previously prepared model to train new one for related problem.

Person 3

[illegible] and [illegible] which are the [illegible] neural networks commonly used for an image classification [illegible]

The use of non [illegible] technology enriches the originality of the solution.

Person 4

Global efficiency and accuracy may depend on neural network architecture.

Weather classification is complicated due to the difficulty of accurately characterizing weather phenomena.

Person 5

Person 6

Person 7

Person 8

3

Group ideas

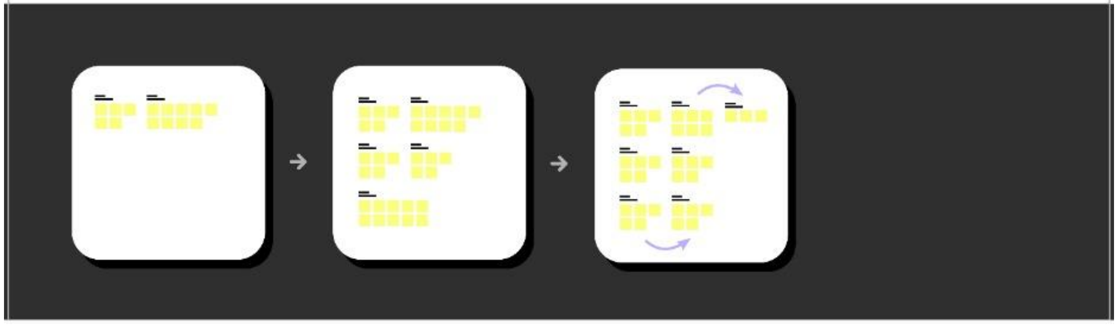
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

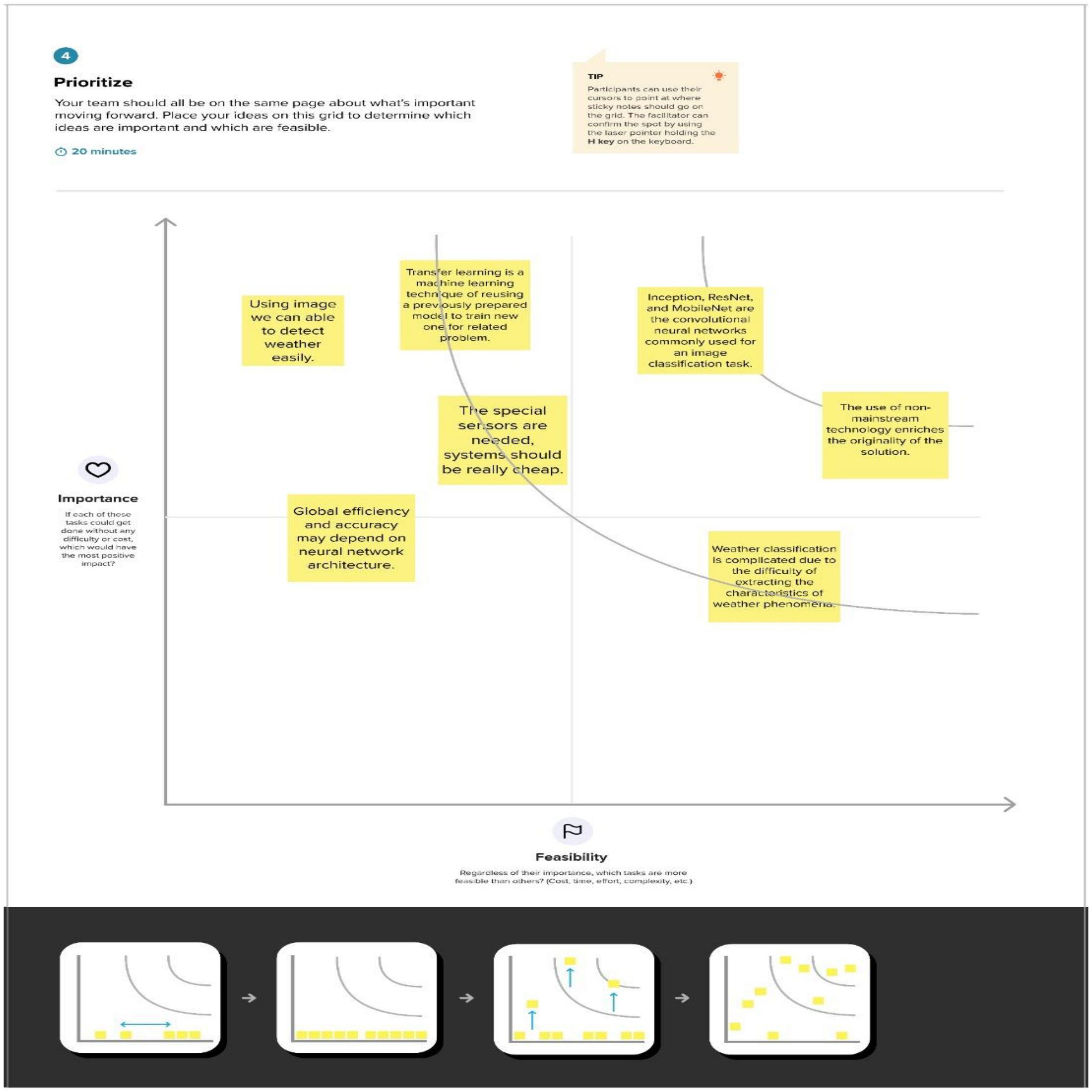
TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Our overall idea about automated weather classification using transfer learning is that we try detect the weather by uploading image in the system and gonna know the weather.



Step-3:IdeaPrioritization



4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

Using image we can able to detect weather easily.

Transfer learning is a machine learning technique of reusing a previously prepared model to train new one for related problem.

Inception, ResNet, and MobileNet are the convolutional neural networks commonly used for an image classification task.

The special sensors are needed, systems should be really cheap.

The use of non-mainstream technology enriches the originality of the solution.

Global efficiency and accuracy may depend on neural network architecture.

Weather classification is complicated due to the difficulty of extracting the charactedstics of weather phenomena.

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

Feasibility

Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

2.3 ProposedSolution:

S.No.	Parameter	Description
1.	ProblemStatement(Problemto besolved)	Weather recognition is a common problem for many branches of industry. Modern agriculture is also based on the analysis of current meteorological conditions. One of the solutions may be a system detecting weather from image. Because any special sensors are needed, the system should be really cheap.
2.	Idea/Solutiondescription	Weather detection using image.
3.	Novelty/Uniqueness	Next-generation radar systems (dual-polarization radar, phased-array radar)
4.	SocialImpact/ CustomerSatisfaction	considers the vulnerability of people and property to the weather and warns of the associated impacts, as well as the likelihood of them occurring.
5.	Business Model(RevenueModel)	A B2C business where data is supplied free and revenue is from advertising, a B2B business where weather data and related insight are delivered to organisations and a bespoke business.
6.	Scalabilityofthe Solution	Global efforts to bring about crucial improvements in supercomputing efficiency and energy usage were placed center stage this week as the European Centre for Medium-Range Weather Forecasts (ECMWF) welcomed users and vendors from around the world to London for the Cray.

3. REQUIREMENTANALYSIS

3.1 Functionalrequirement:

Followingarethefunctionalrequirementsoftheproposedsolution.

FR No.	FunctionalRequirement(Epic)	SubRequirement(Story/Sub-Task)
FR-1	User Registration	Registration through FormRegistration through GmailRegistrationthroughLinkedIN
FR-2	User Confirmation	ConfirmationviaEmail ConfirmationviaOTP
FR-3	User Interface	It allows users to capture images of garbage and seetheresultsoftheclassificationinreal-time.

FR-4	AI Model	TheprojectshoulduseanAIalgorithmthatcan learn fromdataandimproveovertime.
FR-5	Real-time Classification	It should be able to classify images quickly andaccuratelyassoonastheyarecapturedby acamera.

3.2 Non-Functionalrequirements:

Followingarethe non-functional requirementsoftheproposed solution.

FR No.	Non-FunctionalRequirement	Description
NFR-1	Performance	Performance forecasting is an essential service to support decision-taking in the concept, design and operational phases of an asset, meeting the production efficiency challenge by enhancing operational performance.
NFR-2	Reliability	A seven-day forecast can accurately predict the weather about 80 percent of the time

NFR-3	Security	Stay indoors and move to a shelter
NFR-4	Scalability	Global efforts to bring about crucial improvements in supercomputing efficiency and energy usage were placed center stage this week as the European Centre for Medium-Range Weather Forecasts (ECMWF) welcomed users and vendors from around the world to London for the Cray
NFR-5	Usability	Weather Forecasting is crucial since it helps to determine future climate changes.

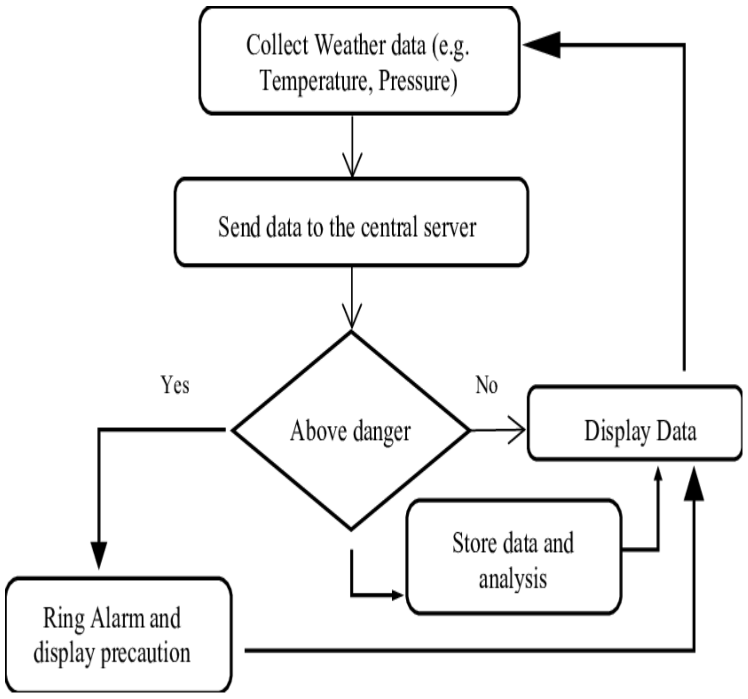
4. PROJECTDESIGN

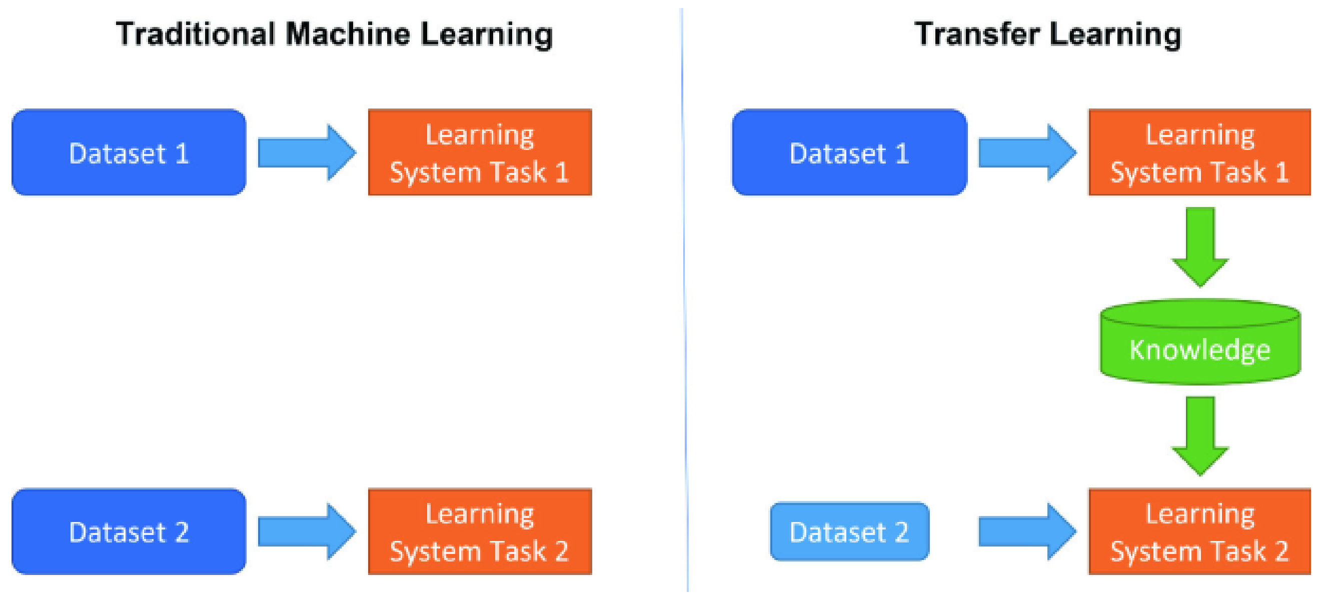
4.1 Data Flow Diagrams

:DataFlowDiagrams:

A Data Flow Diagram (DFD) is a traditional visual representation of the informationflows within a system. A neat and clear DFD can depict the right amount of the systemrequirement graphically. It shows how data enters and leaves the system, what changestheinformation,andwhere dataisstored.

FlowDiagrams:-


Collect Weather data (e.g. Temperature, Pressure)
Send data to the central server
Yes
Above danger
No
Display Data
Store data and analysis
Ring Alarm and display precaution

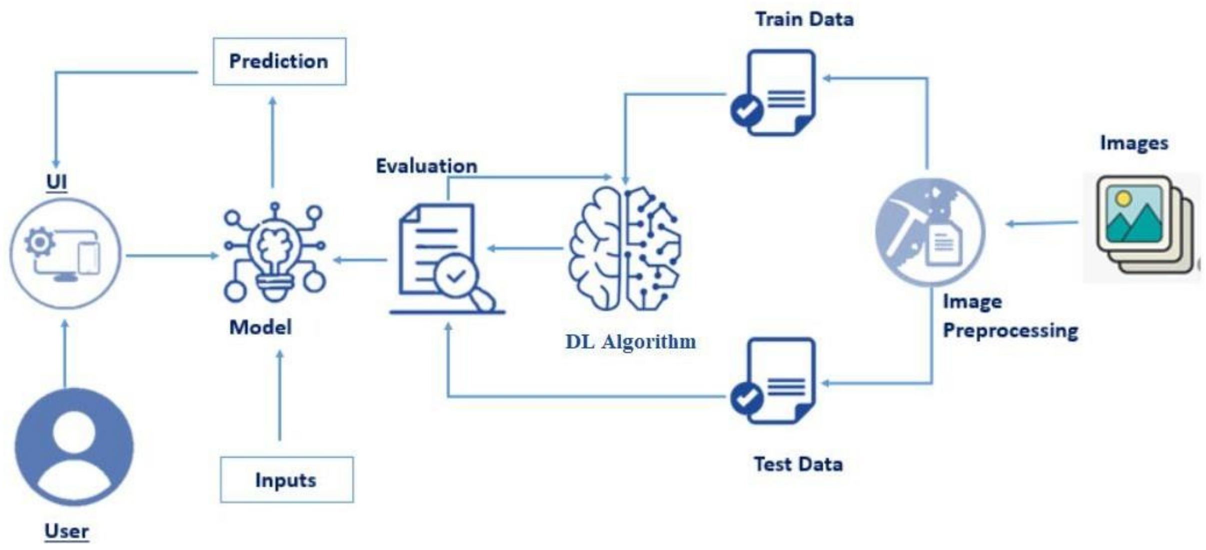


Description:

weather forecasting, Prediction of the weather through application of the principles of physics and meteorology. Weather forecasting predicts atmospheric phenomena and changes on the Earth's surface caused by atmospheric conditions (snow and ice cover, storm tides, floods, etc.).

4.2 Solution&TechnicalArchitecture

Solution Architecture:SolutionArchitectureDiagram:



SolutionArchitecture:
Solution architecture is a complex process – with many sub-processes – that bridgesthegapbetweenbusinessproblemsandtechnologysolutions.Itsgoalsareto:

Findthebesttechsolutiontosolveexistingbusinessproblems.
Describe the structure, characteristics, behavior, and other aspects of thesoftwareto project stakeholders.
Definefeatures,developmentphases,andsolutionrequirements.
Providespecificationsaccordingtowhichthesolutionisdefined,managed,anddelivered.

1. CODING&SOLUTIONING(Explainthefeaturesaddedintheproject alongwithcode)

Feature 1 :

```
import aiohttp
import asyncio
import uvicorn
from fastai import *
from fastai.vision import *
from io import BytesIO
from starlette.applications import Starlette
from starlette.middleware.cors import CORSMiddleware
from starlette.responses import HTMLResponse, JSONResponse
from starlette.staticfiles import StaticFiles

export_file_url =
'https://drive.google.com/file/d/1hIUpWxaPlBtFIu6UdcVkGFXEPPi2gip2/view?usp=sharing'
export_file_name = 'MCIC_Fastai_Model.pkl'

classes = ['cloudy', 'rain', 'shine', 'sunrise']
path = Path(__file__).parent

app = Starlette()
app.add_middleware(CORSMiddleware, allow_origins=['*'], allow_headers=['X-Requested-
With', 'Content-Type'])
app.mount('/static', StaticFiles(directory='app/static'))


async def download_file(url, dest):
    if dest.exists(): return
    async with aiohttp.ClientSession() as session:
        async with session.get(url) as response:
            data = await response.read()
            with open(dest, 'wb') as f:
                f.write(data)


async def setup_learner():
    await download_file(export_file_url, path / export_file_name)
    try:
        learn = load_learner(path, export_file_name)
        return learn
    except RuntimeError as e:
        if len(e.args) > 0 and 'CPU-only machine' in e.args[0]:
            print(e)
            message = "\n\nThis model was trained with an old version of fastai and will not work in a CPU environment.\n\nPlease update the fastai library in your training environment and export your model again.\n\nSee instructions for 'Returning to work' at https://course.fast.ai."
            raise RuntimeError(message)
        else:
            raise


loop = asyncio.get_event_loop()
tasks = [asyncio.ensure_future(setup_learner())]
learn = loop.run_until_complete(asyncio.gather(*tasks))[0]
loop.close()


@app.route('/')
async def homepage(request):
    html_file = path / 'view' / 'index.html'
    return HTMLResponse(html_file.open().read())


@app.route('/analyze', methods=['POST'])
async def analyze(request):
    img_data = await request.form()
    img_bytes = await (img_data['file'].read())
    img = open_image(BytesIO(img_bytes))
    prediction = learn.predict(img)[0]
    return JSONResponse({'result': str(prediction)})


if __name__ == '__main__':
    if 'serve' in sys.argv:
        uvicorn.run(app=app, host='0.0.0.0', port=5000, log_level="info")
```

Html:

```
<html lang='en'>
<head>
  <meta charset='utf-8'>
  <link rel='stylesheet' href='../static/style.css'>
  <script src='../static/client.js'></script>
</head>
<body>
<div>
  <div class='center'>
    <div class='title'>Static Weather Image Classification</div>
    <p>
      Use still image of one of these weather types: <strong>cloudy</strong>,
<strong>rain</strong>, <strong>shine</strong>, <strong>sunrise</strong>
    </p>
    <div class='content'>
      <div class='no-display'>
        <input id='file-input'
               class='no-display'
               type='file'
               name='file'
               accept='image/*'
               onchange='showPicked(this)'>
      </div>
      <button class='choose-file-button' type='button' onclick='showPicker()'>Select
Image</button>
      <div class='upload-label'>
        <label id='upload-label'>No file chosen</label>
      </div>
      <div>
        <img id='image-picked' class='no-display' alt='Chosen Image' height='200'>
      </div>
      <div class='analyze'>
        <button id='analyze-button' class='analyze-button' type='button'
onclick='analyze()'>Analyze</button>
      </div>
      <div class='result-label'>
        <label id='result-label'></label>
      </div>
    </div>
  </div>
</div>
</body>
</html>
```

Feature 2:

```
var el = x => document.getElementById(x);

function showPicker() {
  el("file-input").click();
}

function showPicked(input) {
  el("upload-label").innerHTML = input.files[0].name;
  var reader = new FileReader();
  reader.onload = function(e) {
    el("image-picked").src = e.target.result;
    el("image-picked").className = "";
  };
  reader.readAsDataURL(input.files[0]);
}

function analyze() {
  var uploadFiles = el("file-input").files;
  if (uploadFiles.length !== 1) alert("Please select a file to analyze!");

  el("analyze-button").innerHTML = "Analyzing...";
  var xhr = new XMLHttpRequest();
  var loc = window.location;
  xhr.open("POST", `${loc.protocol}//${loc.hostname}:${loc.port}/analyze`,
    true);
  xhr.onerror = function() {
    alert(xhr.responseText);
  };
  xhr.onload = function(e) {
    if (this.readyState === 4) {
      var response = JSON.parse(e.target.responseText);
      el("result-label").innerHTML = `Result = ${response["result"]}`;
    }
    el("analyze-button").innerHTML = "Analyze";
  };

  var fileData = new FormData();
  fileData.append("file", uploadFiles[0]);

  xhr.send(fileData);
```

}

Inception, ResNet, and MobileNet are the convolutional neural networks commonly used for an image classification task. Although they carry out similar problems and are based on different architectures, some differences can be expected in the results of specific tasks such as weather classification.

3.1 Inception

Inception architecture is based on two concepts - 1 × 1 Convolution and Inception Module. Deep neural networks are expensive in terms of computation. Thanks to 1 × 1 Convolution it is possible to decrease number of computations by reducing number of input channels. It causes that depth and width of neural network can be increased. Inception Module performs computations of some convolution layers simultaneously and then combines results.

InceptionV3 is a convolutional neural network that is 48 layers deep.

The network has an image input size of 299 × 299.

3.2 MobileNet

MobileNet targets mobile and embedded systems. This architecture is based on an inverted residual structure, which connections are between the bottleneck layers. It uses lightweight depthwise convolutions for features filtering.

This architecture allows to build lightweight models which do not need much computing power.

MobileNetV2 is a convolutional neural network that is 53 layers deep.

The he network has an image input size of 224 × 224
3.3 ResNet

ResNet (Residual Networks) uses concept of identity shortcut connection that

allows to jump over some layers. It partially solves vanishing gradients and miti-

gate accuracy saturation problem. The identity shortcuts simplifies the network

and speeds learning process up.

ResNet50 is a convolutional neural network that is 50 layers deep.

The network has an image input size of 224 × 224.

4 Metrics

4.1 Precision, Recall, Accuracy, F1

These metrics are widely used in binary classification where only two categories

are taken into consideration. In multi classification solutions they might be calcu-

lated in multiple ways, but the most popular is to calculate them as the average

of every single metric across all classes. Precision represents proportion of pre-

dicted positives that are truly positive. Values closer to 1 means high precision

and shows that there is a small number of false positives.

P recision = T rueP ositives

T rueP ositives + F alseP ositives

Recall is calculated as a proportion of actual positives that have been classified

correctly. Values closer to 1 means high recall and shows that there is a small

number of false negatives.

Recall = T rueP ositives

T rueP ositives + F alseNegatives

Accuracy measures proportion of number of correct predictions to total number of samples. It helps to detect over-fitting problem (models that overfit have usually an accuracy of 1).

Accuracy = CorrectP redictions

T otalP redictions

F1 Score combines precision and recall metrics by calculating their harmonic mean.

F1=2 ∗ P recision ∗ Recall

P recision + Recall

4.2 Log-Loss, Log-Loss Reduction

Logarithmic loss quantifies the accuracy of a classifier by penalizing incorrect classifications. This value shows uncertainty of prediction using probability estimates for each class in the dataset. Log-loss increases as the predicted probability diverges from the actual label. Maximizing the accuracy of the classifier causes minimizing this function.

Logarithmic loss reduction (also called reduction in information gain - RIG) gives a measure of how much improves on a model that gives random prediction. Value closer to 1 means a better model.

4.3 Confusion Matrix, Micro-averages, Macro-averages

Confusion matrix contains precision and recall for each class in multi-class classification problem.

A macro-average computes the metric independently for each class and then take the average (treats all classes equally).

A micro-average aggregates the contributions of all classes to compute the average metric.

Micro- and macro-averages may be applied for every metric.

In a multi-class classification problem, micro-average is preferred because there might be class imbalance (significant difference between number of class' examples).

5 Dataset

Models were build on custom six class weather image dataset. Images were scraped from web. Despite the fact that the used trainer does not require data normalization [9], the images were normalized to specified aspect ratio (1:1) and size (512 × 512 pixels). This image size has been chosen in order to not to favor any architecture (InceptionV3 prefers 299 × 299, MobileNetV2 and ResNet50 prefer 224 × 224). Training set details are described below.

Total: 1577 images

Image format: JPEG

Image size: 512 × 512 pixels

Color space: sRGB

Categories:

– Clouds

– Fog

– Rain

– Shine

– Storm

– Sunrise

Importance of Weather Forecasting:.
There are various uses of weather forecasting in day-to-day life, it can be as simple as deciding whether to take an umbrella with you on your work or to deciding your outfit. Following are some of the places where weather forecasting plays a major role:Seasons and nature play a major role in agriculture and farming. When it comes to the farming of various fruits, vegetables, and pulses, temperature is extremely important. Farmers didn't have a better understanding of weather forecasts before, so they had to rely on estimates to do their jobs. They do, however, sometimes suffer losses as a result of inaccurate weather forecasts. Farmers will now get all of their forecasts on their smartphones, thanks to advances in technology and the use of unique weather forecasting mechanisms. Of course, education in this area is critical, but the majority of the farmer community at this point understands the fundamentals, making it simple for them to use the features.It aids food grain transportation and storage.It aids in the handling of cultural operations such as harrowing, hoeing, etc.It aids in the implementation of livestock protection initiatives.Weather Forecasting is crucial since it helps to determine future climate changes. With the use of latitude, we can determine the probability of snow and hail reaching the surface. We are able to identify the thermal energy from the sun that is exposed to a region. Climatology is the scientific study of climates, which in simple words mean weather conditions over a period. A bunch of studies within atmospheric sciences also takes the help of the variables and averages of short-term and long-term weather conditions accumulated. Climatology is different from meteorology and can be divided into further areas of study. Different approaches to this segment can be taken. Currently, our primary research goal is to motivate and help the development of efficient and effective measures of Environmental activities.

2. ADVANTAGES&DISADVANTAGES

2.1 Advantages:

Advantages for weather forecasting in agriculture:

For example, weather forecasting enables you to properly plan your farm operations, such as planting, irrigation, fertilizer application, pruning/weeding, harvesting or livestock mating, since farming and agriculture as a whole chiefly depend on seasons and weather.

2.2 Disadvantages:

The following are the disadvantages of weather forecasting
Model Limitations: Forecasting models can only make predictions based on existing data and are limited by the quality and quantity of that data. Limited Time Frame: Forecasts are usually only accurate for a short time frame, making it difficult to plan ahead

CONCLUSION:

Conclusion. In summary, weather forecasts are increasingly accurate and useful, and their benefits extend widely across the economy

FUTURESCOPE:

In addition to predictions of atmospheric phenomena themselves, weather forecasting includes predictions of changes on the Earth's surface climate. These changes are caused by atmospheric conditions like snow and ice cover, storm tides, and floods.

3. APPENDIX

Source Code

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>What's the weather like?</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.6.2/css/bulma.css" />
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css" />
</head>

<body>
    <section class="hero is-primary is-bold">
        <div class="hero-body">
            <div class="container">
                <h1 class="title">
                    What's the weather like?
                </h1>
            </div>
        </div>
    </section>
    <section class="section">
        <div class="container">
            <div class="columns">
                <div class="column is-offset-4 is-4">
                    <form method="POST">
                        <div class="field has-addons">
                            <div class="control is-expanded">
                                <input class="input" name="city" type="text" placeholder="City Name" style="text-transform:
capitalize;">
                            </div>
                            <div class="control">
                                <button class="button is-info">
                                    Add City
                                </button>
                            </div>
                        </div>
                        {% with messages = get_flashed_messages(with_categories=true) %}
                        {% if messages %}
                            {% for category, message in messages %}
                                {% set message_class = 'is-success' %}

                                {% if category == 'error' %}
                                    {% set message_class = 'is-danger' %}
                                {% endif %}
                                <div class="notification {{ message_class }}">{{ message }}</div>
                            {% endfor %}
                        {% endif %}
                        {% endwith %}
                    </form>
                </div>
            </div>
        </div>
    </section>
    <section class="section">
        <div class="container">
            <div class="columns is-multiline">
                {% for weather in weather_data %}
                <div class="column is-one-third">

                    <div class="box">
                        <article class="media">
                            <div class="media-left">
                                <figure class="image is-50x50">
                                    <img src="http://openweathermap.org/img/w/{{ weather.icon }}.png" alt="Image">
                                </figure>
                            </div>
                            <div class="media-content">
                                <div class="content">
                                    <p>
                                        <span class="title">{{ weather.city }}</span>
                                        <br>
                                        <span class="subtitle">{{ weather.temperature }}° C</span>
                                        <br> {{ weather.description }}
                                    </p>
                                </div>
                            </div>
                            <div class="media-right">
                              <a href=" {{ url_for('delete_city', name=(weather.city)) }}">
                                  <button class="delete"></button>
                              </a>
                            </div>
                </article>
              </div>

            </div>
            {% endfor %}
          </div>
        </div>
        </section>
        <footer class="footer">
        <div class="container">
          <div class="content has-text-centered">

            <a href="https://github.com/jkaethee"><i class="fa fa-github" style="font-size:36px"></i></a>.
           </p>
          </div>
         </div>
        </footer>
    </body>

    </html>
```