Task 1: Hand Gesture Classification with Image Preprocessing

Mentor to be contacted: Shaghayegh Vafi *(s.vafi@stud.uis.no)* **and** Ainaz Rafiei *(a.rafiei@stud.uis.no)*

Goal: The goal of this project is to develop a machine learning model that can classify hand gestures into distinct classes.

Dataset: The database is composed by 10 different hand-gestures (showed above) that were performed by 10 different subjects (5 men and 5 women).

Link to download: https://www.kaggle.com/datasets/gti-upm/leapgestrecog

Instructions:

1. **Data Preprocessing**:
   - Read the dataset
   - Converts the image to grayscale to reduce the complexity (since color information is not essential for gesture classification).
   - Standardize image sizes, for example, 64x64 or 128x128 pixels.
   - Applying filters like Gaussian blur to remove noise.
   - You may do Image augmentation techniques like flipping, rotation, scaling, and brightness adjustment to artificially increase the size of the dataset and improve model generalization. (Optional)
   - You may apply edge detection algorithms to highlight the hand from the background
   - Normalize the images by scaling the pixel values to a range of 0-1 for better model performance.
2. **Exploratory Analysis**
   - To begin this exploratory analysis, first use matplotlib to import libraries and define functions for plotting the data. Depending on the data, not all plots will be made.
3. **Feature Extraction**
   - After preprocessing, feature extraction is used to convert image data into a format that a model can easily interpret.
4. **Model selection**:
   - Different machine learning and deep learning models can be used:
     - Traditional Machine Learning Models: If you're using hand-crafted features, models like SVM (Support Vector Machines) and KNN (K-Nearest Neighbors) can be effective.

        - Convolutional Neural Networks (CNNs): CNNs can directly classify gestures from raw or preprocessed images.
        - Also you must apply different models and compare the results.

5. **Training and Evaluation**
    - Split your dataset into training, validation, and testing sets (typically 80% training, 10% validation, 10% testing).
    - Train the model using the training data.
    - Validate the model using the validation data to tune hyperparameters and prevent overfitting.
    - Evaluate the model on test data to see how it generalizes to unseen hand gestures**.**
    - Plot the training and validation accuracy/loss over the epochs.
6. **Evaluation Metrics**:
    - In addition to accuracy, extend the evaluation metrics to include:
        - Confusion matrix
        - Precision, Recall, and F1-score

Task 2: Multilingual Subjectivity in News Articles

Mentor to be contacted: Shaghayegh Vafi *(s.vafi@stud.uis.no)*

Goal: The task is for systems to distinguish whether a sentence from a news article expresses the subjective view of the author behind it or presents an objective view on the covered topic instead. This is a binary classification task where the model has to identify whether a text sequence (a sentence or paragraph) is subjective or objective.

Dataset: https://gitlab.com/checkthat_lab/clef2024-checkthat-lab/-/tree/main/task2

Instructions

1. **Data Preprocessing**:
   - Read the dataset
   - Lowercasing: Convert all text to lowercase to avoid treating words like "News" and "news" as different tokens.
   - Remove Punctuation and Special Characters: Stripping unnecessary characters that don't carry meaning (commas, exclamation points, etc.).
   - Tokenization: Split text into individual words or tokens. Libraries like nltk, spaCy, or transformers can handle this.
   - Stop Word Removal: Remove common words like "the", "is", and "and" that don't contribute much to the subjectivity of a sentence.
   - Lemmatization or Stemming: Reduce words to their base form (e.g., "running" becomes "run").
2. **Exploratory Analysis**
   - To begin this exploratory analysis, first use matplotlib to import libraries and define functions for plotting the data. Depending on the data, not all plots will be made.
3. **Feature Extraction**
   - You can either use one of the methods to convert the text data into numerical features that a model can process.
   - Bag-of-Words (BoW)
   - TF-IDF (Term Frequency-Inverse Document Frequency)
   - Word Embeddings (Word2Vec, GloVe, FastText
   - Pretrained transformer Transformers (e.g., BERT, GPT)
4. **Model selection**:

   - Different machine learning and deep learning models can be used:
     - Logistic Regression, Support Vector Machines (SVM), Random Forest or XGBoost Can be used with BoW or TF-IDF features.
     - Deep Learning (RNNs, CNNs)
     - Transformers (BERT, RoBERTa, DistilBERT)
     - Also, you must apply different models and compare the results.

5. **Evaluation Metrics**:
   - Extend the evaluation metrics to include:
     - Confusion matrix
     - Precision, Recall, and F1-score for a more detailed analysis of the model performance, especially since this is a binary classification task.
   - Analyze the results across multiple languages and report the findings, focusing on how well the model generalizes to multilingual data.
6. **Model Training**:
   - Train the model on the training split and validate on the development test split.
   - Plot the training and validation accuracy/loss curves over the training epochs.
   - Experimenting with different models like BERT or SVMs to see which performs best on your dataset.

Task 3: Face Emotion Detection

Mentor to be contacted: Ainaz Rafiei *(a.rafiei@stud.uis.no)*

Goal: goal is to detect and classify emotions from facial expressions in images or video. Common emotions classified in such models include happy, sad, angry, surprised, neutral, etc.

Dataset: https://www.kaggle.com/datasets/ashishpatel26/facial-expression-recognitionferchallenge

Instructions

1. **Data Preprocessing**:
   - Read the dataset
   - Load the **Haar Cascade face detector** to detect faces in an image
   - Converts the image to grayscale to reduce the complexity
   - Standardize image sizes, for example, 64x64 or 128x128 pixels.
   - Applying filters like Gaussian blur to remove noise.
   - Normalize the images by scaling the pixel values to a range of 0-1 for better model performance
2. **Exploratory Analysis**
   - To begin this exploratory analysis, first use matplotlib to import libraries and define functions for plotting the data. Depending on the data, not all plots will be made.
3. **Feature Extraction**
   - After preprocessing, feature extraction is used to convert image data into a format that a model can easily interpret.
4. **Model selection:**
   - Several deep learning models can be applied for face emotion detection:
     - Traditional Machine Learning Models
     - Convolutional Neural Networks (CNNs)
     - Transfer Learning with Pretrained Models
     - Also you must apply different models and compare the results.
5. **Training and Evaluation**

- Split your dataset into training, validation, and testing sets (typically 80% training, 10% validation, 10% testing).
- Train the model using the training data.
- Validate the model using the validation data to tune hyperparameters and prevent overfitting.
- Evaluate the model on test data to see how it generalizes to unseen images
- Plot the training and validation accuracy/loss over the epochs.

6. **Evaluation Metrics:**
   - In addition to accuracy, extend the evaluation metrics to include:
     - Confusion matrix
     - Precision, Recall, and F1-score

Task 4: Detecting Dementia Using Natural Language Processing

Mentor to be contacted: Shaima Ahmad Freja ***(sa.freja@stud.uis.no)***

Goal: This project aims to develop a predictive model for detecting dementia using Natural Language Processing (NLP). The model will utilize linguistic features extracted from text data to predict dementia diagnoses. You will be responsible for preprocessing the data, performing exploratory analysis, and building machine learning or deep learning models to classify transcripts into either dementia or control categories (binary classification). Students are encouraged to explore model optimization and fine-tuning techniques using frameworks like Scikit-learn, TensorFlow, Keras, Transformer or PyTorch.

Dataset: The dataset for this project is sourced from DementiaBank. The transcript files have been prepared in CSV format and divided into training and testing sets:

- Training Dataset: Includes two files, Dementia_db.csv (containing dementia patient transcripts) and Control_db.csv (containing healthy control transcripts).
- Testing Dataset: Contains one file, testing_db.csv, for evaluating the model's performance.

Link to dataset:

Control_db

Dementia_db

Testing_db

Instructions

1. **Data Preprocessing**:
    - Read transcription files from Training & Testing folders into a dataframe
    - In Testing folder we have transcription files and one text file which contains (label , MMSE) for each transcript file
    - Read lables for the testing dataset from Test_result_label.txt
    - Merging two dataframe (df_Testing & df_label) in one dataframe where filename in the df_testing equal ID in the df_label
        - df_testing: dataframe for texting which contains all the columns except label & mmse
        - df_label: dataframe for testing which contains ID = Filename and lable, mmse columns
    - cleaning the data (lowercase all the texts, remove punctuation marks, all the symbol, ...)
    - Add filter to select only 'ProbableAD', 'PossibleAD' from Dementia data

- Remove punctuation mark
- lemmatization, we will activate removing the stopword here
- Convert the categorical columns like gender to numerical where (female :0, male: 1)
- Save the preprocessing Dataset into csv file with

2. **Exploratory Analysis**
   - To begin this exploratory analysis, first use matplotlib to import libraries and define functions for plotting the data. Depending on the data, not all plots will be made.
3. **Feature Extraction**
   - You can either use one of the methods to convert the text data into numerical features that a model can process.
   - Bag-of-Words (BoW)
   - TF-IDF (Term Frequency-Inverse Document Frequency)
   - Word Embeddings (Word2Vec, GloVe, FastText
   - Pretrained transformer Transformers (e.g., BERT, GPT)
4. **Model selection:**

You can explore traditional machine learning models, deep learning models, or transformer models (optional):

- To Explore traditional machine learning models for text classification such as (Logistic Regression, Support Vector Machine (SVM), Random Forest,...)
- You may also explore deep learning models such as Long Short-Term Memory Networks (LSTM) or Gated Recurrent Units (GRUs)
- Transformer-based models like BERT (Bidirectional Encoder Representations from Transformers) or RoBERTa.

5. **Evaluation Metrics:**
   - Confusion matrix
   - Precision, Recall, and F1-score for a more detailed analysis of the model performance, especially since this is a binary classification task.
6. **Model Training:**
   - Train the model on the training split and validate on the development test split.
   - Plot the training and validation accuracy/loss curves over the training epochs.

Experimenting with different models like BERT or SVMs to see which performs best on your dataset.

Task 5: BBC News Article Classification

Mentor to be contacted: Shaima Ahmad Freja ***(sa.freja@stud.uis.no)***

Goal: This project involves classifying BBC news articles into one of five categories (business, sports, politics, technology, or entertainment). You will be expected to clean and preprocess the data, perform exploratory analysis, and build a machine learning or deep learning model for text classification making it a multiclass classification problem.

Students are encouraged to explore model optimization and fine-tuning techniques using frameworks like Scikit-learn, TensorFlow, Keras, Transformer or PyTorch.

Dataset: The BBC News Dataset contains 2,225 articles from the BBC news website corresponding to stories in five topical areas from 2004-2005. The articles are categorized into five topics: business, sports, politics, technology, and entertainment.

Link to dataset: BBC News Dataset

Instructions

7. **Data Preprocessing**:
    - Read data
    - cleaning the data (lowercase all the texts, remove punctuation marks, all the symbol, ...)
    - show how many categories are available in the dataset
8. **Exploratory Analysis**
    - To begin this exploratory analysis, first use matplotlib to import libraries and define functions for plotting the data. Depending on the data, not all plots will be made.
9. **Feature Extraction**

You can either use one of the methods to convert the text data into numerical features that a model can process.

- Bag-of-Words (BoW)
- TF-IDF (Term Frequency-Inverse Document Frequency)
- Word Embeddings (Word2Vec, GloVe, FastText
- Pretrained transformer Transformers (e.g., BERT, GPT)

10. **Model selection:**
    - Different machine learning and deep learning models can be used:

        - Logistic Regression, Support Vector Machines (SVM), Random Forest or XGBoost Can be used with BoW or TF-IDF features.
        - Deep Learning (RNNs, CNNs)
        - Transformers (BERT, RoBERTa, DistilBERT)
        - Also, you must apply different models and compare the results.

11. **Evaluation Metrics:**
    - Extend the evaluation metrics to include:
        - Confusion matrix
        - Precision, Recall, and F1-score for a more detailed analysis of the model performance, especially since this is a binary classification task.
12. **Model Training:**
    - Train the model on the training split and validate on the development test split.
    - Plot the training and validation accuracy/loss curves over the training epochs.

Experimenting with different models like BERT or SVMs to see which performs best on your dataset.

Task 6: Sentiment Analysis of US Airline Tweets

Mentor to be contacted: Shaima Ahmad Freja ***(sa.freja@stud.uis.no)***

Goal: The goal of this project is to classify tweets related to US airlines into positive, neutral, or negative sentiments. The students will design and implement a classification model to predict the sentiment of airline-related tweets. They will experiment with various machine learning or deep learning models, then evaluate their performance based on accuracy, precision, recall, and F1-score. Students are encouraged to explore model optimization and fine-tuning techniques using frameworks like Scikit-learn, TensorFlow, Keras, Transformer or PyTorch.

Dataset: The dataset is the US Airline Sentiment Dataset, containing 14,640 tweets labeled as positive, neutral, or negative. The task is to predict the sentiment of each tweet, making it a multiclass classification problem.

Link to dataset: US_Airline_Tweets

Instructions

1. **Data Preprocessing**:
    - Read data
    - cleaning the data (lowercase all the texts, remove punctuation marks, all the symbol, ...)
    - Load the dataset and Check for missing or inconsistent data, then clean the dataset accordingly.
    - Preprocess the text data (Tokenization, Remove stop words, punctuation, Apply stemming or lemmatization (splitting text into individual words or tokens).
    - show how many categories are available in the dataset
2. **Exploratory Analysis**
    - To begin this exploratory analysis, first use matplotlib to import libraries and define functions for plotting the data. Depending on the data, not all plots will be made.
3. **Feature Extraction**

You can either use one of the methods to convert the text data into numerical features that a model can process.

- Bag-of-Words (BoW)
- TF-IDF (Term Frequency-Inverse Document Frequency)
- Word Embeddings (Word2Vec, GloVe, FastText
- Pretrained transformer Transformers (e.g., BERT, GPT)

4. **Model selection:**
    - Different machine learning and deep learning models can be used:
    - You can explore traditional machine learning models, deep learning models, or transformer models (optional):
    - To Explore traditional machine learning models for text classification such as (Logistic Regression, Support Vector Machine (SVM), Random Forest,...)
    - You may also explore deep learning models such as Long Short-Term Memory Networks (LSTM) or Gated Recurrent Units (GRUs)
    - Transformer-based models like BERT (Bidirectional Encoder Representations from Transformers) or RoBERTa..
5. **Evaluation Metrics:**
    - Extend the evaluation metrics to include:
        - Confusion matrix
        - Precision, Recall, and F1-score for a more detailed analysis of the model performance, especially since this is a binary classification task.