Intelligent Sensor Fusion with Deep Learning

Asm Nurussafa
Electronics Engineering
Hochschule Hamm-Lippstadt
Lippstadt, Germany
asm.nurussafa@stud.hshl.de

***Abstract*—Odor classification is very important in both industry and everday life. In this paper, we will introduce sensor fusion and deep learning models to classify products using their odor. We will use data from multiple gas sensors from different products such as tea, coffee etc. We will then use these data and combine them to train a model using Convolutional Neural Network (CNN) and then use this model to predict and evaluate futher data.**

***Index Terms*—Odor classification, Sensor Fusion, CNN, Keras, Edge Impulse.**

I. Introduction

Odor detection and classification is crucial in many fields, from public health to industrial applications. Our sense of smell can alert us to dangers such as gas leaks, fires, and spoiled food, as well as help us identify pleasant scents like flowers, perfumes, and food aromas. One of the most significant benefits of odor detection is its potential for identifying disease. Studies have shown that certain illnesses like Parkinson's and Alzheimer's disease can be detected through changes in body odor, even before clinical symptoms appear. In fact, researchers are currently working on developing ”electronic noses” that can detect specific odor signatures of various diseases, allowing for earlier diagnosis and treatment. [1]
In industry, odor classification plays a crucial role in quality control. For example, in the production of food and beverages, the presence of unpleasant odors can indicate contamination or spoilage. In the perfume industry, the ability to accurately classify and describe scents is essential to creating new fragrances and maintaining consistency in existing products.
In this paper, we will look at a simple study of odor detection to identify different types of tea, coffee etc. and discuss how sensor fusion techniques with deep learning models can be combined to better predict such products.

II. Sensor Fusion

Sensor fusion is the process of combining data from multiple sensors to improve the accuracy and reliability of measurements and inferences. This technique is commonly used in many fields, including robotics, autonomous vehicles, aerospace, and healthcare.

The main idea behind sensor fusion is to compensate for the limitations of individual sensors by combining their outputs into a more accurate and reliable estimate of the measured quantity. For example, if we want to estimate the position and orientation of a robot, we can use multiple sensors such as cameras, gyroscopes, accelerometers, and GPS, each of which provides partial and sometimes conflicting information about the robot's state. By fusing the data from these sensors, we can obtain a more robust and accurate estimate of the robot's position and orientation. [3]. Sensor fusion can be done in various ways, depending on the specific application and the types of sensors involved. There are many sensor fusion techniques [4].

Sensor fusion has many advantages, such as improving the accuracy, robustness, and reliability of measurements, reducing the effects of sensor noise and errors, and providing redundant information in case of sensor failures. However, it also has some challenges, such as the need for accurate calibration and synchronization of the sensors, the selection of appropriate fusion algorithms, and the management of large amounts of data. There are various types of sensor fusion [5] [6] , including:

- Data-level fusion: combining raw sensor data at the signal level.
- Feature-level fusion: extracting features from the raw data before combining them.
- Decision-level fusion: combining the decisions made by multiple sensors or algorithms.
- Hybrid fusion: combining multiple levels of fusion or combining multiple types of sensors.

III. Convolutional Neural Network (CNN)

A convolutional neural network (CNN) is a type of deep neural network that is widely used for image and video processing tasks. It is inspired by the structure and function of the visual cortex in the human brain, which is specialized in detecting and recognizing visual patterns. The key feature of CNNs is their ability to learn spatial hierarchies of features from the input data. They achieve this by using convolutional layers that apply filters or kernels to the input data and produce feature maps that capture the presence of certain visual patterns at different scales and locations in the input. [7] [8]

CNNs typically consist of multiple layers, including convolutional, pooling, and fully connected layers, and are trained using backpropagation and stochastic gradient descent. They are capable of learning complex and abstract features from the input data, and have achieved state-of-the-art performance on many computer vision tasks, such as image classification, object detection, segmentation, and recognition. Some of the advantages of CNNs include their ability to handle high-dimensional and complex input data, their ability to learn and generalize from large amounts of training data, and their ability to reduce the number of parameters and computational complexity compared to fully connected networks. [9] [10]

IV. Keras

Keras is an open-source deep learning library that is widely used for building and training various types of neural networks, including convolutional neural networks (CNNs). It provides a simple and user-friendly interface for designing, configuring, and training deep learning models, and supports both CPU and GPU computation. In the context of CNNs, Keras provides a high-level interface for defining the architecture of the network, including the number and type of layers, the activation functions, and the optimization algorithm. It also provides built-in functions for loading and preprocessing image data, as well as for visualizing and evaluating the performance of the model.
To use Keras for building a CNN, the user first needs to define the model architecture, typically using the Sequential or Functional API. The model can then be trained using the fit() function, which takes the input data, the output labels, and various training parameters such as batch size, number of epochs, and validation split. During the training process, Keras uses backpropagation and gradient descent to update the parameters of the model and minimize the loss function. It also supports various optimization algorithms such as stochastic gradient descent (SGD), Adam, and RMSprop, and allows the user to customize the learning rate and other hyperparameters. Once the model is trained, it can be used for making predictions on new data using the predict() function, and the performance can be evaluated using various metrics such as accuracy, precision, recall, and F1 score. [11] [12]

V. Case Discussion

A. Obtaining raw dataset

For our case study, we will work with the dataset from Benjamin Cabe [13]. The data is takes from odors of tea, coffee and some spirits. Benjamin used the SeeedStudio's Multichannel Gas Sensor v2 which can identify carbon monoxide(CO), nitrogen di-oxide(NO2), ethanol and some amounts of different volatile compounds. To augment this system, a spg30 sensor was added which gives some equivalent carbon dioxide mixture and a different measure of volatile organic compounds and finally, to run some tesets, a BME 680 sensor was used that can measure temperature, humidity , pressure and some air quality. These are then connected to a easy to use Wio Terminal (an Arduino-compatible prototyping platform). With this data from these sensors, we can simply use machine learning algorithms that can automatically find correlation between the features and combine them to classify different types of odors. However, this only works if we are simply trying to make an overall prediction and classification

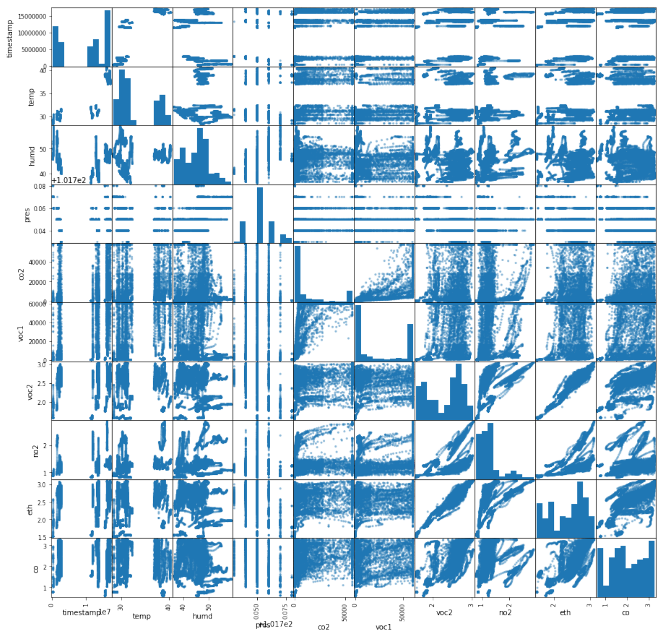

Fig. 1. Features of all raw data.

about somethings's odor. If we need the actual accurate gas measured, this technique likely would not work at least for the limited data we are planning to gather.

The data is then collected from three different environments: kitchen, office and outdoors.

B. Uploading dataset and Pre-processing the data

We will then upload the dataset from the different environments to a python script in Google Colab environment. The following plot, Fig. 1 shows the feature from all the different products in different environment.

So, the data are taken from the following products: background, coffee, spirit-gin, spirit-rum, spirit-vodka, spirit-whiskey, tea-black, tea-chamomile.
The features, as seen from the figure, are: temperature (temp), humidity (hum), carbon dioxide (co2), volatile compound 1 (voc1), volatile compound 2 (voc2), nitrogen dioxide (no2), ethanol (eth) and lastly, carbon monoxide (co). A correlation between the features can be seen in Fig. 2. As seen, the yellow boxes show that correlation is 1, as expected. The darker the color gets, the less correlation it has to that particular feature.

Acknowledgment

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression "one of us (R. B. G.) thanks ...". Instead, try "R. B. G. thanks...". Put sponsor acknowledgments in the unnumbered footnote on the first page.

References

[1] Polymenidou, M., and Sklaviadis, T. (2018). Odor Detection and Disease Diagnosis. In Molecular Diagnostics (pp. 137-144). Humana Press.
[2] Wilson, A. D., and Baietto, M. (2010). Applications and advances in electronic-nose technologies. Sensors, 10(2), 470-498.

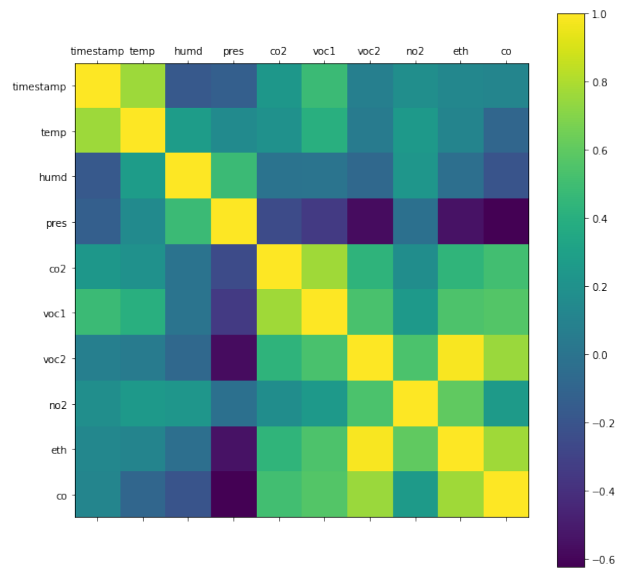

Fig. 2. Correlation between the features.

[3] Durrant-Whyte, H., and Bailey, T. (2006). Simultaneous localization and mapping: part I. IEEE Robotics and Automation Magazine, 13(2), 99-110.

[4] Borenstein, J., Everett, H., and Feng, L. (1996). Where am I? Sensors and methods for mobile robot positioning. University of Michiga

[5] Liu, H., Hu, X., Li, W., Xu, M., and Jiang, L. (2017). A survey of multisensor fusion methods. Information Fusion, 35, 68-75.

[6] Khan, S., and Ullah, S. (2017). A review of data fusion techniques. Information Fusion, 33, 100-111.

[7] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[8] Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep learning. MIT Press.

[9] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[10] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., and Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[11] Chollet, F. (2015). Keras: Deep learning library for Theano and TensorFlow. https://keras.io/

[12] Brownlee, J. (2019). Introduction to Convolutional Neural Networks with Keras for Machine Learning. Machine Learning Mastery. https://machinelearningmastery.com/introduction-to-convolutional-neural-networks-with-keras-for-machine-learning/

[13] https://blog.benjamin-cabe.com/2021/08/03/how-i-built-a-connected-artificial-nose