# KAPITEL 4 / *CHAPTER 4* [4]
# IMBALANCED DATA: A COMPARATIVE ANALYSIS OF CLASSIFICATION ENHANCEMENTS USING AUGMENTED DATA

**DOI: 10.30890/2709-2313.2024-28-00-017**

## Introduction

In the modern world, the volume of available data is increasing every year, but in many cases, the quality and quantity of data may be insufficient for effective training of various machine learning and artificial intelligence models. Therefore, to improve the performance and accuracy of these models, there is a need to develop effective data augmentation methods.

Data augmentation is the process of creating new data based on existing data, which helps to improve the representation of various aspects and characteristics of the data. The importance of data augmentation is high in various fields such as computer vision, speech recognition, and data analytics, and it can play a crucial role in building accurate and efficient models [1].

However, challenges in data augmentation prevent achieving high accuracy and performance of machine learning and artificial intelligence models. Without adequate understanding and application of effective data augmentation methods for different modalities, there is a high likelihood of encountering problems such as:

Overfitting: Models may become too specific to the training data and fail to generalize their predictions to new data.

Insufficient data: Lack of sufficient quantity and quality of data can result in poor training results and inadequacy of models for solving practical tasks.

Time and computational costs: Without using optimal data augmentation methods, model training can become time-consuming and computationally expensive.

All these issues can lead to low-quality results, wasted efforts in model development, and inefficient use of computational resources. This underscores the importance of research and development of data augmentation methods for different

[4] ***Authors:*** *Melnykova Nataliia, Paterega Iurii*

modalities, which will help address the aforementioned problems and achieve better results in building and applying machine learning and artificial intelligence models [2,3].

Therefore, data augmentation methods help increase the volume and diversity of training data, providing better model generalization and reducing the risk of overfitting. However, data augmentation typically depends on the data modality, so it is important to research and understand which augmentation methods are effective for different modalities, such as tabular data, images, and audio data.

## 4.1. Current housing condition

### *4.1.1. Classification*

Classification is one of the types of supervised learning in the field of machine learning, the goal of which is to recognize (classify) instances or samples into one of several predefined classes or categories based on their features or characteristics.

The classification task involves constructing a set of functions or rules that can map instances of input data, represented as feature vectors, to corresponding class labels. These functions are designed to capture the underlying structure and relationships in the data to distinguish and separate classes [4,5].

Mathematically, this can be represented as follows:

Input Data: The input data for the classification problem typically consists of a vector $X = (x1, x2, ..., x_n)$, containing numerical or categorical values representing features or attributes of an object or observation. This vector belongs to the feature space denoted as X.

Output Data (Classes): The output results of the classification problem are a discrete variable Y, which has a set of predefined categories or classes $C = \{C1, C2, ..., C_k\}$. The goal is to find a function (algorithm or model) that assigns a certain class C to one of the input vectors X.

Dataset: A dataset D is provided, which typically contains a series of observations

or samples (instances). Each instance is represented by a feature vector Xi and its corresponding class label Yi, denoted as (Xi, Yi) for i = 1, 2, ..., N, where N is the total number of instances in the dataset.

The classification problem aims to evaluate the function f: X → Y so that for any input instance x, f(x) predicts the correct class label y for that instance. Ideally, the function f should be able to minimize the classification error of the data instances while maintaining an acceptable level of complexity [6].

In machine learning, the classification process looks as follows:

Data Collection and Preprocessing: Obtain a dataset containing feature vectors and their corresponding class labels. Preprocessing may involve cleaning, normalization, or feature engineering to ensure data quality and suitability for modeling.

Feature Selection: Determine the most relevant features that significantly contribute to class separation and discrimination. This step can help improve model performance and reduce computational complexity by removing irrelevant or redundant features.

Model Selection: Then, choose an appropriate classifier based on the data characteristics and problem requirements. Examples of classification algorithms include logistic regression, support vector machines (SVM), decision trees, random forests, k-nearest neighbors (KNN), and neural networks.

Model Training: The labeled dataset is divided into training and validation (or testing) sets. Use the training set to "train" the classifier on the relationships between input features and class labels by adjusting its internal parameters. Depending on the chosen algorithm, this step may involve optimizing the cost function, constructing decision boundaries, or learning feature weights.

Model Evaluation: After training, assess the trained model using the validation set, which contains unseen instances for evaluating model performance and its generalization ability. Evaluation metrics such as accuracy, precision, recall, and f1-score can be used to quantitatively assess the classifier's success in predicting correct class labels.

Additionally, during instance classification, we may encounter the problem of insufficient data. The absence of data in the classification process can significantly affect the performance and efficiency of classification models. A limited dataset not only poses challenges in model development but can also lead to certain issues during the classification process.

Some problems that may arise due to a limited data sample include:

Overfitting: With a limited number of training data, models may overfit, meaning they start memorizing training samples instead of learning to generalize data patterns. As a result, models may perform well on training data but fail to provide accurate predictions on unseen data.

Underrepresentation: A smaller dataset may not cover the entire feature space and adequately represent the classes, resulting in a poor model. Such models may fail to capture the true characteristics of the problem and may be ineffective or biased in their predictions.

Class Imbalance: In scenarios where the dataset is small and the class distribution is skewed, the model training process will be biased towards the majority of classes. This leads to poor prediction performance for minority classes, as the model is insufficiently exposed to their distinct features and patterns.

Variability and Uncertainty: A small dataset may lack sufficient diversity of samples, leading to high variability and uncertainty in model predictions. As a result, the performance of any particular test dataset may be far from the expected overall performance, resulting in unreliable results.

One way to address the problem of insufficient data is data augmentation.

### *4.1.2. Data augmentation*

Data augmentation is a technique used in machine learning, especially in the context of deep learning, to expand and enrich the training dataset by creating new data instances using various transformations applied to existing data. These transformations are intended to preserve the original class labels while introducing variations and diversity in the data that can mimic real-world scenarios [7,8].

In detail, data augmentation performs several tasks to help address the problem of data scarcity:

Increasing the dataset size: By applying various transformations to the original instances in the training dataset, data augmentation creates a larger set of diverse samples. These additional data help reduce the risk of overfitting and allow models to learn and extract more discriminative features for improved generalization to unseen data [9,10].

Enhancing data diversity: The diverse variations introduced through data augmentation effectively expand the feature space and provide a more comprehensive representation of the data distribution. By training on a more diverse dataset, models become more robust and adaptable to real-world situations, resulting in higher prediction accuracy.

Addressing class imbalance: Data augmentation can help alleviate class imbalance by selectively creating more instances for underrepresented classes. This balances the class distribution and facilitates better learning of class-specific features, ultimately improving prediction performance across all classes.

Transformation invariance: Data augmentation through operations such as rotation, scaling, and flipping allows models to learn to be robust to these transformations and maintain prediction accuracy despite such changes in real-world data instances.

Implicit regularization: Data augmentation also serves as a form of implicit regularization as it encourages models to learn reliable and invariant features rather than memorizing training data. This can help reduce overfitting and improve model generalization to new data instances.

Depending on the data modality, different artificial data augmentation techniques can be employed.

### *4.1.3. Classification algorithms for different data modalities.*

Choosing the appropriate machine learning algorithm for classification tasks depends on various factors, including the data modality, dataset size, feature

representation, computational resources, and desired performance metrics. Different data modalities, such as images, text, audio, and numerical data, exhibit unique characteristics and require specialized feature extraction and learning methods [11,12].

4.1.3.1 Classifier for text data.

One of the machine learning methods for classifying text data is Logistic Regression [21]. It is a widely used supervised machine learning algorithm for binary and multiclass classification tasks. It is a generalized linear model (GLM) that models the probability of a certain outcome using the logistic function. Logistic regression is particularly useful when working with categorical dependent variables and continuous or discrete independent variables.

Logistic regression can be an effective method for text classification tasks, especially in cases with limited computational resources or when a simple, interpretable model is required. Some advantages of logistic regression for text classification are:

Multidimensional and sparse data. Text data, after feature extraction, typically leads to multidimensional and sparse representations, where each dimension corresponds to a unique word or lemma from the vocabulary. Logistic regression performs well in such situations as it can handle a large number of features while maintaining a relatively simple linear model that can be interpreted.

Scalability. Logistic regression is computationally efficient and can scale well with the size of the dataset and the number of features. This makes it a suitable choice for text classification tasks, especially when dealing with limited computational resources or when a fast classification model is needed.

There are several strategies for extending logistic regression for multiclass classification problems, let's consider them:

One vs Rest (OVR) or One vs All (OVA) strategy. In this approach, we train multiple binary logistic regression classifiers, one for each class. For each classifier, we consider one class as positive (target class) and the rest as negative (non-target class). To make a prediction, we pass a test instance through each classifier and choose the class corresponding to the classifier with the highest probability score.

One vs One (OVO) strategy. In the "one vs one" strategy, we train a binary logistic regression classifier for each pair of classes, resulting in N·(N-1)/2 classifiers for N classes. Classifiers are trained only on instances from the corresponding class pairs. To predict for instance, we consider the results of all classifiers and choose the class that receives the most "votes" from individual classifiers.

Softmax Regression. Softmax is an extension of logistic regression for multiclass classification problems. Instead of training multiple binary classifiers, we directly model the probability of each class. The softmax function, also known as the normalized exponential function, is applied to the linear output results to obtain a probability distribution over the target classes. Like binary logistic regression, we use the maximum likelihood estimation (MLE) principle to estimate the model parameters.

4.1.3.2 Image Classifier.

A Convolutional Neural Network (CNN) [13] is a type of deep learning model specialized in processing grid-like data, such as images, using convolutional layers to capture local spatial patterns and abstractions. CNNs have become the primary model for numerous image classification tasks due to their ability to learn hierarchical features, noise resilience, and variations in input data.

The key layers of CNNs for image classification are:

Input Layer: The input layer receives the raw image data. Each input image is typically represented as a three-dimensional tensor (width, height, and number of color channels - usually 3 for RGB images).

Convolutional Layers: These layers perform convolution operations, which involve applying a set of trainable filters to the input image or the output of previous layers. During convolution, the filter slides over the image, element-wise multiplying the filter with the local neighborhood and then summing the results to create a feature map. Each filter detects specific features or patterns in the input data, such as edges, lines, textures, or more complex structures at deeper network levels.

Non-linearity (Activation functions): Activation functions introduce non-linearity into the network, allowing it to learn and approximate complex, non-linear functions. The most commonly used activation function is the Rectified Linear Unit (ReLU),

which preserves positive values and sets negative values to zero.

Pooling Layers: Pooling layers reduce spatial dimensions and computational complexity, making the model more computationally efficient and invariant to changes. The most commonly used pooling technique is Max-Pooling, which takes the maximum value from a certain local neighborhood (usually 2x2) and moves across the image.

Fully Connected Layers: Fully connected layers are used to map the results from the convolutional and pooling layers into a final class prediction or probability distribution. In a typical architecture, there may be one or more fully connected layers, followed by a softmax activation function to output probabilities for each class.

Output Layer: The output layer produces the final class probabilities for image classification. For multiclass tasks, softmax activation is often used, while for binary classification, a sigmoid activation function can be used. The class with the highest probability is chosen as the final network prediction.

CNNs can also be further enhanced using the following layers:

Batch Normalization: This is a technique commonly used in CNNs for faster convergence and better generalization. It involves normalizing the output of a layer using learned scaling and shifting parameters to ensure a stable, predefined mean and standard deviation.

Dropout: To reduce overfitting and improve generalization, dropout layers are sometimes added to CNN architectures. Dropout layers randomly "drop out" a certain percentage of neurons during training, forcing the network to learn more robust features.

Regularization: This is also an important technique in machine learning aimed at preventing overfitting by adding a penalty term to the loss function. Overfitting typically occurs when the model learns to capture noise in the training data, leading to poor generalization to unseen data. In convolutional neural networks (CNNs), regularization methods help improve generalization and increase the stability of the training process.

All these techniques are used to construct a model that performs well with various

input data. Employing these principles in conjunction with data augmentation is a good practice leading to better generalization, improved model performance, and increased robustness to variations in input data.

4.1.3.3. Audio Classifier.

Audio classification is the process of categorizing and labelling different types of audio signals into separate classes or categories based on their content or characteristics. The primary goal of audio classification is to automatically analyse, understand, and organize large volumes of audio data. Several components and methods play a crucial role in developing an effective and reliable sound classification system. One of them is the feature extraction stage. The feature extraction stage is usually the first step in the task of audio classification. Initially, relevant features are extracted from the raw audio signal. The choice of features significantly impacts the classification effectiveness, and depending on the specific application, different types of features may be used. Commonly used features include spectrograms, Mel-Frequency Cepstral Coefficients (MFCCs), chroma features, and spectral contrast.

Mel-Frequency Cepstral Coefficients (MFCCs) [14] are a widely used set of features in audio signal processing, particularly for speech and music analysis. They were first introduced in the 1980s as a means of representing the spectral characteristics of audio signals compactly and efficiently. The main idea behind MFCCs is to closely mimic the human auditory system, which tends to be more sensitive to certain frequency ranges than others.

The calculation of MFCCs involves several key steps:

Preprocessing: The raw audio signal is first preprocessed by applying a high-pass filter to enhance high-frequency components and reduce noise effects. This is done to emphasize frequencies that are more relevant for distinguishing speech sounds.

Framing and Windowing: The audio signal is then divided into small overlapping frames, typically around 20-40 milliseconds each. This is done to capture local characteristics of the audio signal, assuming that the spectral properties of the signal are relatively stable within such a short duration. A windowing function, usually a Hamming or Hanning window, is applied to each frame to minimize side lobes effects

in the frequency domain.

Fourier Transform and Power Spectrum: The Discrete Fourier Transform (DFT) is applied to each windowed frame to transform it into the frequency domain. This results in a complex-valued spectrum and phase spectrum for each frame. The square magnitude (or power) spectrum is then computed, forming the basis for further analysis.

Mel Filtering: To obtain a frequency scale that better corresponds to human auditory perception, a bank of Mel filters is applied to the power spectrum. The Mel scale is a logarithmic frequency scale that attempts to approximate human perception of frequency differences. The Mel filterbank typically consists of overlapping triangular filters uniformly spaced on the Mel scale and covering the entire frequency range of the power spectrum. The output of each filter represents the energy present in the corresponding Mel-frequency bin.

Logarithm and Discrete Cosine Transform (DCT): A logarithmic function is applied to the output of the Mel filterbank to obtain logarithmic energy values. This step approximates human perception of sound intensity. Subsequently, the Discrete Cosine Transform (DCT) is applied to the logarithmic energy values to obtain the cepstrum. DCT helps decorrelate the spectral information, creating a compact representation. Several first cepstral coefficients (usually between 12-20) are retained as the final MFCC features, as these coefficients capture the most relevant information about the spectral envelope.

Feature extraction is an important step in audio classification as it helps transform raw data into a meaningful and compact representation suitable for analysis and classification tasks.

For audio classification, we will use a Multilayer Perceptron (MLP) [15]. This type of artificial neural network with a feedforward architecture consists of multiple layers of interconnected neurons. It is a supervised learning algorithm that uses labelled training data to learn the relationship between input features and corresponding output labels. MLP is widely used for various classification and regression tasks. Neurons in an MLP are connected, and each connection has a weight that determines the strength

of the connection between neurons. During the training process, these weights are adjusted using a learning algorithm, such as backpropagation and gradient descent, to minimize the error in predicting output labels.

The basic structure of a perceptron consists of the following layers:

Input Layer: The input layer is responsible for receiving input features and passing them to the next layer. The number of neurons in this layer corresponds to the number of features in the input data.

Hidden Layers: Hidden layers are located between the input and output layers. MLP can have one or multiple hidden layers. Neurons in hidden layers process and transform the information received from the previous layer and pass the result to the next layer. The complexity of the learned function increases with the number of hidden layers and the number of neurons per layer.

Output Layer: The output layer produces the final output for a given input. The number of neurons in the output layer depends on the type and complexity of the problem. For binary classification, the output layer typically has one neuron with a sigmoid activation function, while for multiclass classification, the output layer uses a softmax activation function with one neuron for each class.

**4.2. Results**

Overview of datasets and their augmentation. The software implementation was done using the Google Colaboratory cloud environment and the Python programming language.

Text Data Augmentation.

For investigating the impact of augmentation on textual data, a dataset regarding customer reviews on Amazon products and the sentiment of these reviews were selected.

Let's take a look at the dataset. (Figure 1.)

The dataset contains 17,340 rows and consists of the following columns:

• Sentiments. The target column of the dataset - the sentiment of the customer review. It can have one of three values: positive, negative, or neutral sentiment.

| | sentiments | cleaned_review | cleaned_review_length | review_score |
|---|---|---|---|---|
| 0 | positive | i wish would have gotten one earlier love it a... | 19 | 5.0 |
| 1 | neutral | i ve learned this lesson again open the packag... | 88 | 1.0 |
| 2 | neutral | it is so slow and lags find better option | 9 | 2.0 |
| 3 | neutral | roller ball stopped working within months of m... | 12 | 1.0 |
| 4 | neutral | i like the color and size but it few days out ... | 21 | 1.0 |
| ... | ... | ... | ... | ... |
| 17335 | positive | i love this speaker and love can take it anywh... | 30 | 5.0 |
| 17336 | positive | i use it in my house easy to connect and loud ... | 13 | 4.0 |
| 17337 | positive | the bass is good and the battery is amazing mu... | 41 | 5.0 |
| 17338 | positive | love it | 2 | 5.0 |
| 17339 | neutral | mono speaker | 2 | 5.0 |

**Figure 1. Dataset of product reviews**

• Cleared_review. The main feature of the dataset - the text of the review.

• Cleared_review_length. The length of the review.

• Review_score. Product rating.

In this case, the classifier will be trained to predict the sentiment of the reviews. Therefore, we are only interested in the Sentiments and Cleared_review columns. Additionally, text preprocessing is required. The text needs to be converted to lowercase and stop words need to be removed. After preprocessing, the dataset looks as follows. (Figure 2.)

| | text |
|---|---|
| 0 | wish would gotten one earlier love makes worki... |
| 1 | learned lesson open package use product right ... |
| 2 | slow lags find better option |
| 3 | roller ball stopped working within months mini... |
| 4 | like color size days return period hold charge |
| ... | ... |
| 17335 | love speaker love take anywhere charge phone w... |
| 17336 | use house easy connect loud clear music |
| 17337 | bass good battery amazing much better charge t... |
| 17338 | love |
| 17339 | mono speaker |

**Figure 2. Dataset of product reviews**

Let's examine the class balance in the dataset (Figure 3.).

From the graph, it is evident that the majority of classes are positive and neutral reviews, while negative reviews constitute a significant minority. This could negatively

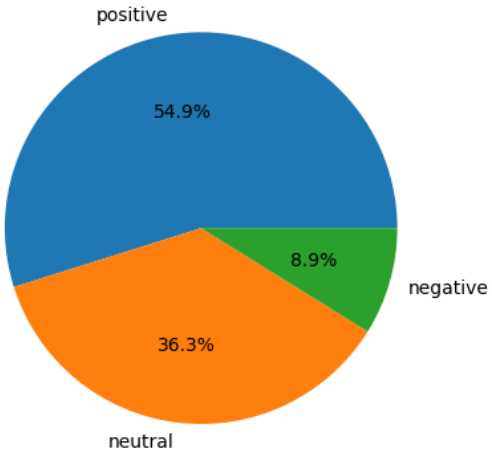


**Figure 3. Class distribution in the dataset**

impact the training process and the final predictions of the model, so augmentation would be beneficial for this dataset.

Let's consider how the data will look after augmentation using various methods to increase textual data. Each augmentation method was applied twice to a sentence. We will use the textaugment library and its EasyDataAugmentation class for this purpose, which provides various functions for sentence manipulation.

• Synonym replacement. We can observe that the new sentences contain different synonyms similar in meaning to the old ones.

| text | synonym_replacement |
|---|---|
| looking wireless rechargeable mouse recently lost old mouse kind glad really love one super cute... | looking wireless rechargeable mouse recently lost old mouse kind glad really love one super cute... |
| head set good comfortable head good gaming | head band practiced comfortable head practiced gaming |
| sound reflects lot fooled image | speech sound reflects wad fooled image |
| good sound quality durable | good voice quality perdurable |
| product used gaming | intersection put upon gaming |
| wish missed return window one mouse range meters stated difficulty tracking meter even couple fe... | wish missed return window unity mouse range meters stated difficulty tracking meter even couple ... |
| control shift work saw older reviews supposed new version keyboard hoped fixed nope features bac... | control shift work saw older reviews hypothetic new version keyboard hoped fixed nope features b... |
| cute color lights works great | precious color visible radiation works great |
| bought mouse color led quite cool would expect work like normal mouse clicking one click quite h... | bought mouse color precede quite cool would expect work like normal mouse sink in one click quit... |
| type connector bent unable use way get replacement type connector | type connector bent ineffectual utilisation way get replacement type connector |

**Figure 4. Dataset augmented using synonym replacement method**

• Random insertion. New words appear at random positions in the sentence.

| text | random_insertion |
|---|---|
| battery lasted hrs fully charged recommend returning | battery lasted repay hour hrs fully charged recommend returning |
| bought best friend go gaming laptop christmas last year loved worked even survive til one | bought best sour friend go gaming laptop christmas last year sour loved worked even survive til one |
| thought keep charging work hour days died yet charged time week | thought keep charging work hour twenty four hours days died yet charged break time week |
| love noise cancellation comfortable | love noise cancellation dear devout comfortable |
| never right review help people save money time buy product sucks bad takes min type freezing | never right review help people save money time buy product sucks bad supporter takes min case ty... |
| great mouse especially price led lights awesome works perfect easily connects device | great mouse especially price led lights awing awesome works perfect easily awing connects device |
| sorry usually dont leave bad reviews one takes cake started working first started laggy longer w... | sorry usually dont leave bad reviews one takes cake started working first started laggy bug out ... |
| good worked used basic office work nothing heavy scroll button broke despite almost never using ... | good worked nigh used basic office work nothing heavy scroll button broke despite almost billet ... |
| sound quality far exceeds price speaker battery life exceptional bluetooth connectivity could ea... | sound quality speaker system far exceeds price speaker battery life alir exceptional bluetooth c... |
| charge week love would recommend product super easy set | charge week urge love would recommend product slowly super easy set |

**Figure 5. Dataset augmented using random insertion method**

• Random swaps. Words in the sentence change their positions.

| text | random_swap |
|---|---|
| cheaply made another get pay suggest pass | get cheaply another made pay suggest pass |
| lasted last months | last months lasted |
| great bass rich amazing sound | sound rich bass amazing great |
| broke weeks poorly put together | put broke poorly weeks together |
| packaging damaged looked used | looked damaged used packaging |
| less months purchasing scrolling wheel stopped working | months less working scrolling wheel stopped purchasing |
| sensitive mouse maybe new works fine | sensitive maybe works new mouse fine |
| mouse cool comfortable use fan sleep mode annoying still like mouse love color shifting feature | annoying cool comfortable use fan sleep mode mouse feature like mouse love color shifting still |
| always wanted mouse similar apple one found use windows laptop stylish rgb light looks pretty co... | like wanted mouse saving apple one found use windows laptop stylish rgb light looks pretty cool ... |
| love mine hear house | love house mine hear |

**Figure 6. Dataset augmented using random swaps method**

Image Augmentation.

For image augmentation, a dataset of lung X-ray images with and without pneumonia was selected. It consists of 5863 X-ray images (JPEG) and two categories (pneumonia/normal). The chest X-ray images were selected from a medical center in Guangzhou. All chest X-ray examinations were conducted as part of routine clinical care for patients. For the analysis of chest X-ray images, all chest X-rays were initially screened for quality control by removing any scans of low quality or unreadable.

Example images from the dataset (Figure 7.)

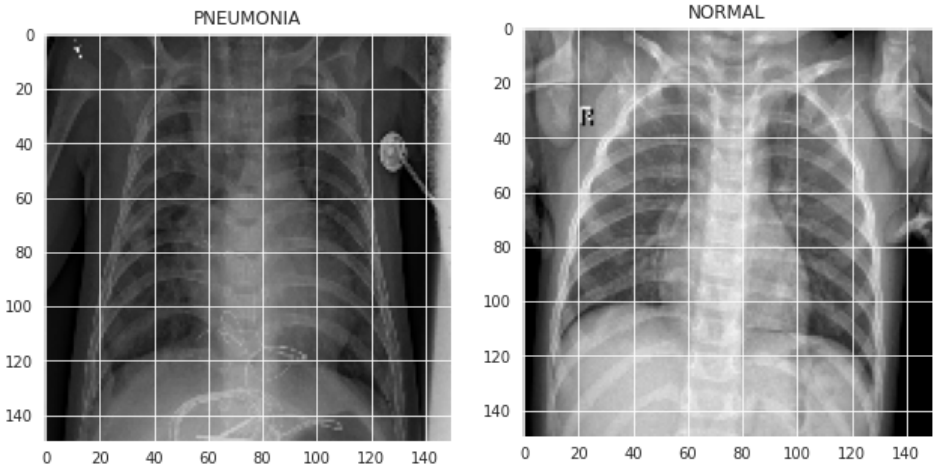


**Figure 7. Some images from the dataset**

Let's look at the class balance in the image dataset (Figure 8.).

In this situation, the scenario is similar. The class of images with the disease predominates over healthy lungs. With the use of augmentation, we will balance the classes.

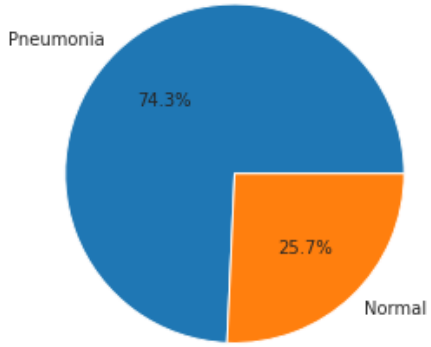


**Figure 8. Class distribution in the image dataset**

To do this, we will utilize various image augmentation techniques. We will use the ImageDataGenerator from the keras library for this purpose. It works by applying various transformations to the images in your dataset, such as scaling, rotation, zooming, and flipping. The ImageDataGenerator augments data on the fly during each epoch, providing batches to the neural network during training. This ensures that the model sees different variations of the same image in different epochs.

Let's consider how the data will look after augmentation using different combinations of parameters to increase the number of images:

• Combination of image rotations. In this case, images will be randomly flipped horizontally or vertically. Additionally, they may be rotated at a certain angle.

```
model_rotation, history_rotation = run_test(x_train, y_train, x_val, y_val,
                                            rotation_range = 30,
                                            horizontal_flip = True,
                                            vertical_flip=True)
```

**Figure 9. Combination of augmentation parameters with rotations**

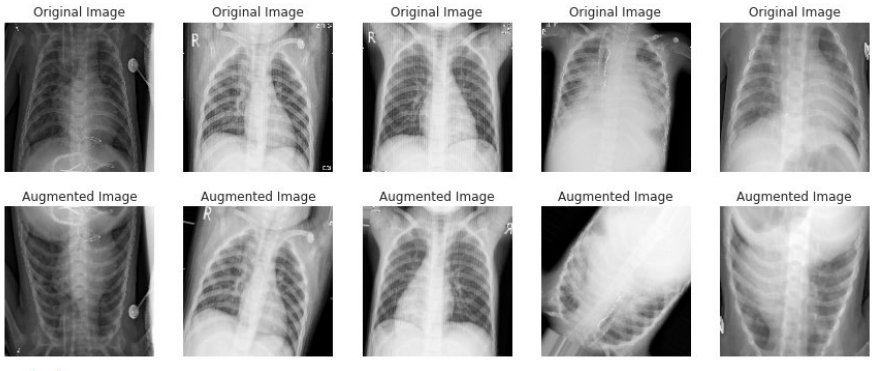


**Figure 10. Images augmented with rotations**

• Combination of zooming and height or width shifting of images. In this case, images will be randomly zoomed in or out. Additionally, they may be arbitrarily shifted in width or length from the overall fraction.

```
model_zoom, history_zoom = run_test(x_train, y_train, x_val, y_val,
                                    zoom_range = 0.2,
                                    width_shift_range=0.1,
                                    height_shift_range=0.1
                                   )
```

**Figure 11. Combination of augmentation parameters with zooming and shifts**

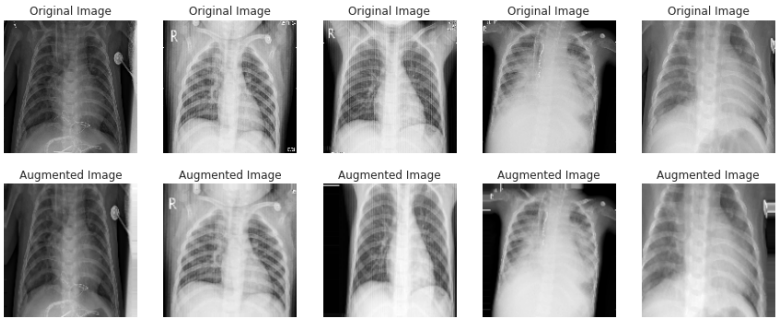


**Figure 12. Images augmented with zooming and shifts**

• Combination of brightness change. In this case, images will randomly change the brightness of pixels.

```
model_brightness, history_brightness = run_test(x_train, y_train, x_val, y_val,
                                                brightness_range=(0.5, 1.2),
                                                zoom_range = 0.2
                                                )
```

**Figure 13. Combination of augmentation parameters with brightness change**

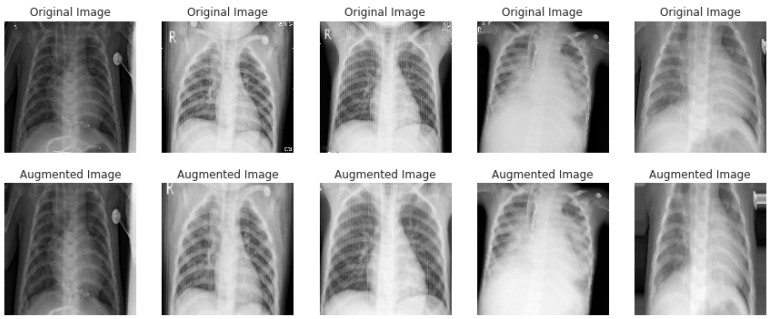


**Figure 14. Images augmented with brightness change**

### 4.3. Discussion

Experiments were conducted for three data modalities using various augmentation methods. After applying augmentation, an analysis of the model's effectiveness and the quality of augmented data using the metrics mentioned above was conducted. The results obtained using augmentation were compared with the results obtained on the original dataset. For each comparison, tables will be constructed for each modality, demonstrating the impact of augmentation and the percentage improvement in metrics.

The change in results will be calculated using the formula:

$$Variation = ((m\_augmented\ dataset) - m\_original\ dataset) * 100\% \quad (1)$$

The tables show the percentage improvement (green color) or deterioration (red color) of the results:

• Text

**Table 1 Change in Text Data Metrics**

| Augmentation / Metric | Synonym Replacement | Random Swaps | Random Insertions |
|---|---|---|---|
| Accuracy | 0.9% | 5.1% | 4.9% |
| Precision | -6.9% | 5.7% | 3.4% |
| Recall | 13.8% | 11.4% | 11.7% |
| F1-score | 9.2% | 12.8% | 12.5% |

We observe that synonym replacement had the least impact on the results, with a decrease in precision. This could be due to the replacement of certain synonyms affecting the sentence's tone and leading to incorrect results. Meanwhile, augmentation using random swaps or insertions proved beneficial for this dataset, significantly improving prediction quality.

Image

**Table 2 Change in Image Dataset Metrics**

| Аугментація / Метрика | Випадкові повороти | Випадкові зсуви | Випадкові зміни яскравості |
|---|---|---|---|
| Accuracy | 14% | 19% | -11% |
| Precision | 3% | 8% | -53% |
| Recall | 21% | 26% | -15% |
| F1-score | 22% | 27% | -26% |

Significant deteriorations were observed with brightness changes. This is because X-ray images are highly sensitive to such alterations, sometimes resulting in loss of pneumonia data or falsely identifying pneumonia in healthy images. However, augmentation using rotations and shifts proved highly effective for this dataset, leading to significant performance improvements.

• Audio

**Table 3 Change in Audio Dataset Metrics**

| Augmentation / Metric | Synonym Replacement | Random Swaps | Random Insertions |
|---|---|---|---|
| Accuracy | 6.5% | 8.2% | 7.3% |
| Precision | 5.2% | 6.5% | 5.8% |
| Recall | 6.9% | 8.77% | 7.81% |
| F1-score | 7% | 8.76% | 7.86% |

Augmentation for the audio dataset yielded similar results across all methods. The employed methods were well-suited for this dataset and did not affect the dataset's essential features.

It is essential to choose augmentation methods according to the dataset characteristics to achieve the best results. Augmented data should not be excessively distorted, as this may lead to incorrect class predictions. Therefore, it is crucial to judiciously adjust parameters controlling the random degree of data variation and select methods based on the data type. Improperly chosen methods can significantly degrade data quality and, consequently, results. For instance, in textual data, the context of some synonyms may vary, leading to content distortion. For grayscale images, changes in brightness or contrast can have negative effects, resulting in the formation of white or dark spots after augmentation. Hence, it is necessary to experiment with optimal methods and parameters, combining some augmentation methods to identify the best approaches for adding artificial data to address specific classification problems.

**Summary and conclusions**.

The paper discusses aspects of classification applied to different data modalities, including text, images, and audio. The classification challenges in the context of working with imbalanced data are highlighted, the consequences of which can be mitigated by applying data augmentation methods for underrepresented classes. Various classifiers, from logistic regression to convolutional neural networks, can be used in classification depending on several factors. The importance of evaluating model quality using different performance metrics to determine their effectiveness in relevant scenarios was described.

Experiments were conducted for three data modalities using three different augmentations for each modality. The experiments were carried out on the original dataset and the augmented ones. Subsequently, a comparative analysis of prediction improvements was conducted to assess the impact of each method on prediction quality.