CytoAutoCluster: A Semi-Supervised Approach to Cell Classification

Introduction

In data-driven environments, assessing the features of objects and classifying them into distinct groups is critical for various applications. CytoAutoCluster focuses on clustering cells into different groups based on their features. The objective is purely computational, aiming to develop an efficient clustering system using semi-supervised learning techniques.

Problem Overview

The challenge of clustering cytometry data arises primarily due to the following reasons:

- **Lack of Labels**: Most of the available cytometry data is unlabeled, making it difficult to directly classify the cells into groups.
- **Data Collection Difficulty**: It is challenging to obtain sufficient data that explicitly defines which group a cell belongs to.
- **Complex Features**: High-dimensional data from cytometry requires advanced algorithms to capture and process relevant features for classification.

Objectives

The aim of the CytoAutoCluster project is to design a clustering algorithm capable of classifying cells into distinct groups based on their features using a semi-supervised learning approach. This involves:

- Learning from a limited amount of labeled data to identify the most important features.
- Applying these features to classify a large volume of unlabeled data.
- Developing a computational system that works efficiently.

Approach: Semi-Supervised Learning

Since obtaining labeled data is resource-intensive, semi-supervised learning provides an optimal approach by leveraging both labeled and unlabeled data. The strategy includes:

- **Labeled Data**: A small fraction of the dataset has labels indicating the groupings of cells based on their features.
- **Unlabeled Data**: The majority of the dataset remains unlabeled, and the model must infer groupings from the labeled data.

Through this method, the system can learn significant patterns from the labeled data and apply these to the unlabeled portion for efficient classification.

Dataset Selection

Various datasets were considered for this project, but the following were key:

1. **Rejected Dataset**:
   CellCnn Learning Disease-Associated Cell Subsets
   Reason for Rejection: This dataset was rejected as it pertains to medical applications, which do not align with the project's goal of purely computational classification of cells.
2. **Accepted Dataset**:
   Levine CytOF 32 Dimensional Data
   Reason for Acceptance: This dataset has more than 60% unlabeled data and includes 40 features, making it suitable for semi-supervised learning where the majority of data is unlabeled, and feature-rich data is required.

Code Implementation for CytoAutoCluster

Loading the Dataset:

The project begins with loading the Levine CytOF 32 Dimensional Data dataset into the environment. This dataset is stored in CSV format and is uploaded to Google Colab for analysis.

Exploring the Data:

After loading the dataset, initial exploration is conducted to understand its structure. This includes examining the first few rows and determining the overall shape of the data, which helps in identifying the number of features and instances available for analysis.

Checking for Null Values:

Given the emphasis on utilizing semi-supervised learning, assessing the presence of null values in the dataset is crucial. A thorough analysis is performed to identify which columns contain null values and the specific rows that are affected. This step is essential for understanding the completeness of the data.

Comparison between Null and Non-Null Values:

To gain insights into the distribution of null and non-null values across the dataset, visualizations are created. These graphical representations help to identify which features are impacted by missing values and the extent of this impact, informing subsequent preprocessing decisions.

Data Analysis Techniques:

Histograms: Distribution of values for each feature.
Correlation Matrix: Identifies relationships between features, assessing feature importance**.**

Correlation Matrix:

A correlation matrix is constructed to analyze relationships between features. This matrix helps identify:

- **Positive Correlations**: Features that increase or decrease together.

- **Negative Correlations**: Features where one increases as the other decreases.
- **Independent Features**: Features with minimal correlation, often useful for dimensionality reduction.

Understanding correlations aids in feature selection and helps to reduce redundancy by removing highly correlated features, which simplifies the model.

Range of Feature Values :

Each feature's value range is calculated, providing insights into the scale and variability of each attribute.

Box Plot :

Box plots are used to visually examine the distribution of each feature:

- **Median and Quartiles**: Centered around the median, it shows data spread and skewness.
- **Outliers**: Box plots highlight outliers beyond the interquartile range, guiding further decisions about feature engineering or data cleaning.

Skewness and Kurtosis Analysis:

- **Skewness**: Measures the asymmetry of feature distributions:
  - Positive Skew: Right-skewed, with a longer tail on the right.
  - Negative Skew: Left-skewed, with a longer tail on the left.
- **Kurtosis:** Measures the "tailedness" or peakedness of the distribution:
  - High Kurtosis: Leptokurtic, with sharper peaks and heavier tails.
  - Low Kurtosis: Platykurtic, with flatter distributions.

This analysis helps identify features requiring transformation to improve normality and stabilize variance, often beneficial for machine learning algorithms.

Dimensionality Reduction Techniques:

1. **Principal Component Analysis (PCA)**

   PCA is used to reduce dimensionality by retaining essential features while preserving variance. It simplifies the data for visualization and analysis, especially with high-dimensional data like this project's dataset.

   - Variance Explained: Calculated to assess the significance of each component, which helps retain 95%+ of the variance using fewer dimensions.
   - Transformation to Lower Dimensions: This simplification reduces computational costs and overfitting risks, enhancing model efficiency.

2. **t-Distributed Stochastic Neighbor Embedding (t-SNE)**

   t-SNE is primarily a visualization tool that maps high-dimensional data into 2D or 3D space to reveal underlying clusters and patterns:

   - Preserving Local Similarities: Unlike PCA, t-SNE captures non-linear relationships by preserving local structures rather than global.

- Cluster Visualization: Highlights potential groupings within data, particularly effective in semi-supervised learning for better understanding of unlabeled data structure.

Autoencoders

Autoencoders are neural networks used for feature extraction in semi-supervised learning, capable of handling both labeled and unlabeled data.

Autoencoder Mechanism**:**

- **Encoding**: Compresses input data into latent features.
- **Latent Space**: Stores reduced-dimensional data.
- **Decoding**: Reconstructs input data from latent space, minimizing dimensionality while preserving key information.

Applications and Data Requirements:

1. **Input Dimensions**: Autoencoders for tabular data input high-dimensional cytometry data, adapting it for reduced-dimension representations.
2. **Corruption Techniques**: Artificial noise or dropout can be applied to input data to improve feature extraction.