Universidad Francisco de Vitoria UFV Madrid

Computer Science and Engineering

Second Semester - Course 2024/2025

Artificial Intelligence 2

Done by

Christopher Cobo Piekenbrock

Rodrigo Sáez Escobar

Gonzalo Eusa Silvela

Jaime López de Heredia Delgado

Juan Fernández Cerezo

Teacher

Moisés Martínez Muñoz

Abstract

This practical work focuses on implementing and analyzing a Self-Organizing Map (SOM) to classify Pokemon based on their characteristics. The dataset includes attributes such as Pokemon type, statistics (attack, defense, special attack, special defense), and type effectiveness. The data was carefully prepared through cleaning, transformation, and normalization to optimize the SOM training. We experimented with various configurations to identify the most suitable model for our analysis. Evaluation methods like classification maps, activation maps, quantization errors, and topographic errors were used to assess the model's accuracy and efficiency. Our results show effective clustering of Pokemon, reflecting meaningful groupings according to type and attributes.

Table of Contents

Table of Figures

List of Tables

1. Introduction

The objective of this practical work is to develop a Self-Organizing Map (SOM) for classifying Pokemon based on their attributes Through this implementation, we aim to understand how SOMs work, including their training process, classification, and evaluation. By implementing a SOM, we try to analyse how Pokemon with similar characteristics group together, to understand the relationships between different Pokemon types and their features.

A key part of this project is the preparation and preprocessing of the dataset, which includes attributes such as Pokedex number, name, type, generation, attack, defense, special attack, special defense, and type vulnerabilities. The dataset will be cleaned and transformed to guarantee effective training of the SOM. Also, we will test different configurations for training parameters, including map size, learning rate, and neighborhood function, in order to identify a model that then will be used to answer the questions and the analysis.

The evaluation of the trained SOM will be conducted using classification maps, activation maps, and error measurements such as quantization and topographic errors. These analyses will help determine the effectiveness of the model in correctly classifying Pokemon based on their attributes.

Instead of just trying to get the lowest error values, our goal is to find the best model for our work. This means balancing accuracy, simplicity, and efficiency while making sure the results match our expectations and provide useful insights, not just good numbers.

2. Project Overview

2.1. Description of the Final Dataset

The dataset contains information about Pokemon species, their attributes, and their effectiveness against different types. It is structured as a tabular dataset where each row represents an individual Pokemon entry, and each column represents a specific attribute.

Dataset structure

The dataset consists of the following columns:

1. **Effectiveness Against Other Types**

- *against_bug, against_dark, against_dragon, against_electric, against_fairy, against_fight, against_fire, against_flying, against_ghost, against_grass, against_ground, against_ice, against_poison, against_psychic, against_rock, against_steel, against_water*

- These columns represent the effectiveness multiplier of the Pokemon's type(s) when facing a particular type. A value greater than 1 indicates an advantage, while a value less than 1 indicates a disadvantage.

2. **Base Statistics**

- *attack*: The base attack stat of the Pokemon.
- *defense*: The base defense stat of the Pokemon.
- *sp_attack*: The base special attack stat of the Pokemon.
- *sp_defense*: The base special defense stat of the Pokemon.

Key characteristics

The dataset provides insights into Pokemon strengths, weaknesses, and battle effectiveness. It indicates how each Pokemon performs against different types, allowing for a deeper understanding of type matchups between the Pokemons.

Summary of our final dataset

Variable	Type of data	Range	Example of data	Number of elements
against_bug	float64	0.000 - 1.000	0.200	796
against_dark	float64	0.000 - 1.000	0.200	796
against_dragon	float64	0.000 - 1.000	0.500	796
against_electric	float64	0.000 - 1.000	0.125	796
against_fairy	float64	0.000 - 1.000	0.067	796
against_fight	float64	0.000 - 1.000	0.125	796
against_fire	float64	0.000 - 1.000	0.467	796
against_flying	float64	0.000 - 1.000	0.467	796
against_ghost	float64	0.000 - 1.000	0.250	796
against_grass	float64	0.000 - 1.000	0.000	796
against_ground	float64	0.000 - 1.000	0.250	796

against_ice	float64	0.000 - 1.000	0.467	796
against_poison	float64	0.000 - 1.000	0.250	796
against_psychic	float64	0.000 - 1.000	0.500	796
against_rock	float64	0.000 - 1.000	0.200	796
against_steel	float64	0.000 - 1.000	0.200	796
against_water	float64	0.000 - 1.000	0.067	796
attack	float64	0.000 - 1.000	0.244	796
defense	float64	0.000 - 1.000	0.196	796
sp_attack	float64	0.000 - 1.000	0.299	796
sp_defense	float64	0.000 - 1.000	0.214	796

Table 1. Summary of the final dataset

2.2. Description of the methods used for data preparation

Before training our Self Organizing Map (SOM), we had to prepare the dataset to avoid errors and be sure that it was structured and ready to be used. We have followed several steps to clean, transform and analyze the data.

Once we downloaded the dataset in our Google Colab group environment, we checked its structure to understand the information it contained. We verified how many Pokemon´s were in the dataset, the type of data in each column, and whether any values were missing. This first examination of the data helped us decide how to manage the data correctly.

We noticed that some Pokemon in the dataset did not have a secondary type. Instead of leaving these values empty, we assigned them a placeholder label ("Nan") to be sure that all Pokemon had a value in that column.

Initially, we considered using Label Encoding to convert Pokemon types (type1 and type2) into numerical values. However, after analyzing the dataset, we decided to remove these columns entirely, because all the against attributes already provide the necessary information about strengths and weaknesses.

Additionally, we removed:

Name and pokedex_number since they are unique identifiers that do not contribute to the classification process, generation because it only orders Pokemon based on predefined categories, grouping them together even if they have completely different characteristics and against_normal because most Pokemon types are neutral to normal type attacks, and it was causing clusters to form mainly based on this feature rather than meaningful differences We removed them so the SOM could focus only on the most relevant attributes without being influenced by unnecessary information.

To better understand feature relationships, we created a correlation matrix and visualized it using a heatmap. This analysis helped us confirm which attributes were essential for classification.

We also used a Box Plot to check if there were any outliers in the dataset. At first, we thought about removing them because they could affect the training. But then we realized that removing outliers would mean deleting some Pokemon from the dataset, so removing them would mean losing valuable data. So, in the end, we decided to keep all Pokemon's to make sure the model learns from all the data.

Since Pokemon attributes had different scales, we applied Min-Max Scaling to have the verification that all features had the same range (0 to 1). This normalization was essential because SOMs uses distance-based calculations, and features with larger numerical values like Attack could dominate the clustering process if it was not properly scaled. So, our main reason was having all attributes contributing equally to the clustering. Min-Max Scaling was chosen over Standard Scaling because SOMs works better with values in a fixed range (0-1) rather than having mean centered values.

Finally, to avoid division by zero in cases where the minimum and maximum values were the same, we added a small constant.

By following these steps, our dataset was properly prepared for training, allowing the SOM to focus on relevant attributes.

2.3. Experimentation process

Before training our Self Organizing Map, we need to define some key functions to make sure the model could learn correctly.

We are going to implement the plan that we have studied in class. Firstly, we need to find the best matching unit (BMU). Our function *cal_bmu* finds the neuron in the SOM that is closest to a given input. It does this by calculating the Euclidean distance between the input and each neuron in the weight matrix. In Self-Organizing Maps (SOMs), choosing the right distance measurement is important for comparing data points and neurons. Euclidean distance is usually chosen over Manhattan distance because of its geometric features and usefulness in high-dimensional spaces [1]. It matches the natural way we think about distances, making it a good choice for grouping similar data in SOMs [2]. Euclidean distance also squares the differences in each dimension, giving more weight to larger differences, which helps in creating clearer clusters. Because of its mathematical advantages, it is the standard distance measurement in many SOM applications [1]. As we have seen in the theory class, the neuron with the smallest distance is the BMU. Continuing with the *learning_rate_variation,* that we used to control how much the neuron weights change with each step. It starts high and decreases over time to allow the SOM to stabilize as it learns.

The choice of neighborhood function affects how well the model organizes and separates data. We used the Mexican Hat function instead of the Bubble or Gaussian functions because it helps create clearer groups and better distinctions between features.

The Mexican Hat function works by exciting nearby neurons while slightly farther neurons are weakened. This approach helps form well-defined clusters, making it easier to separate different groups of data [3]. Another reason for choosing the Mexican Hat function is that it follows a pattern similar to real biological neural networks, where close neurons strengthen each other while farther ones are inhibited. This behavior helps the SOM create a more natural and organized representation of the data [4]. The Mexican Hat function is also better at making clusters more distinct. Since it reduces the impact of neurons that are slightly farther from the BMU, it prevents different groups from overlapping, which improves how well SOM separates patterns [5].

The Gaussian and Bubble functions work differently. The Gaussian function smoothly reduces the effect of neurons as they move away from the BMU, but it does not have the inhibition feature of the Mexican Hat function. The Bubble function treats all neurons within a fixed range the same way, which can result in less precise grouping [6].

With the *neighborhood_variation* we can see how much nearby neurons of the BMU are adjusted based on the distance from the BMU. With the *calculate_bmu_influence* we calculate how much a neuron is influenced by the BMU during training.

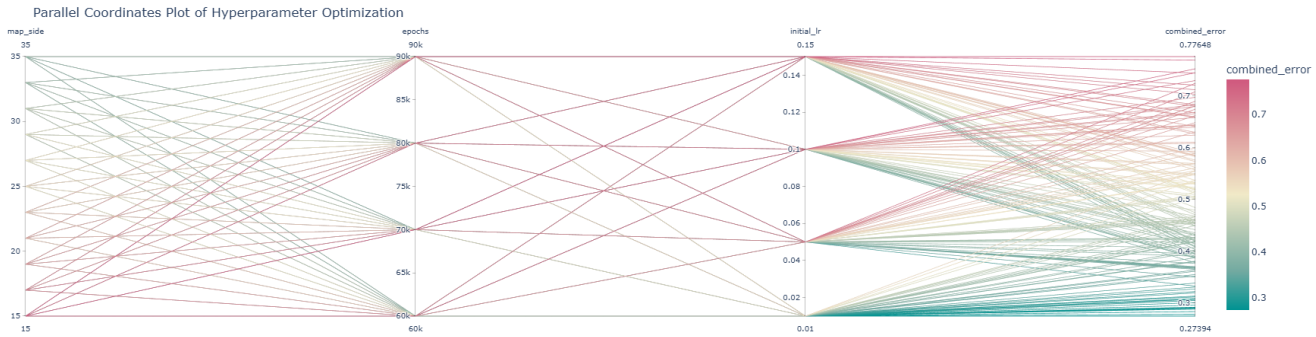

Figure 1 -Experimentation Phase Combinations

Figure 1 presents different combinations of hyperparameters (map size, epochs, and initial learning rate) tested during our experimentation phase. By comparing these configurations, we identified the most effective model that achieved a balance.

Also, we have represented a colour legend to help visualize better the different combined errors using logical colours. Red for higher error and green for less error.

2.4.Answers to the evaluation questions

2.4.1. Evaluation of our best model

Evaluating the performance of our Self-Organizing Map (SOM) is very important to understanding how well it classifies Pokemon's based on their attributes. This section presents various representations and error metrics used to assess the model. By examining different visual representations and performance measures, we can determine the quality of clustering and be sure that the SOM accurately captures relationships within the dataset.

Number of clusters

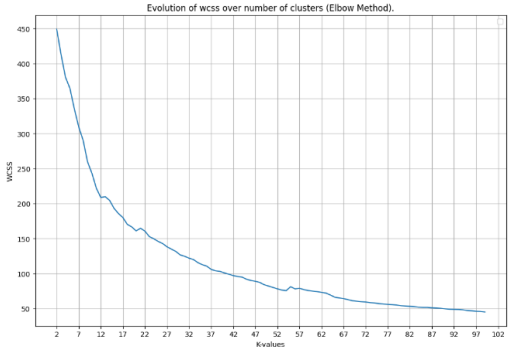

Figure 2-Number of Clusters (Elbow method)

```
The optimal number of clusters (k) is: 12
```

Figure 3-Number of clusters using Kneed library

As it shown in the Figure 2, we applied the elbow method to determine the optimal number of clusters for our model. This method involves plotting the within-cluster sum of squares (WCSS) against the number of clusters and looking for an "elbow point," where the

rate of decrease slows down. However, identifying the exact number of clusters from the graph can sometimes be subjective and challenging. To address this, we utilized the Kneed library, which is an automated way to detect the number of clusters by identifying the knee point mathematically. The number of clusters suggested by Kneed (Figure 2) was then used for some representations later in the document.

Classification map

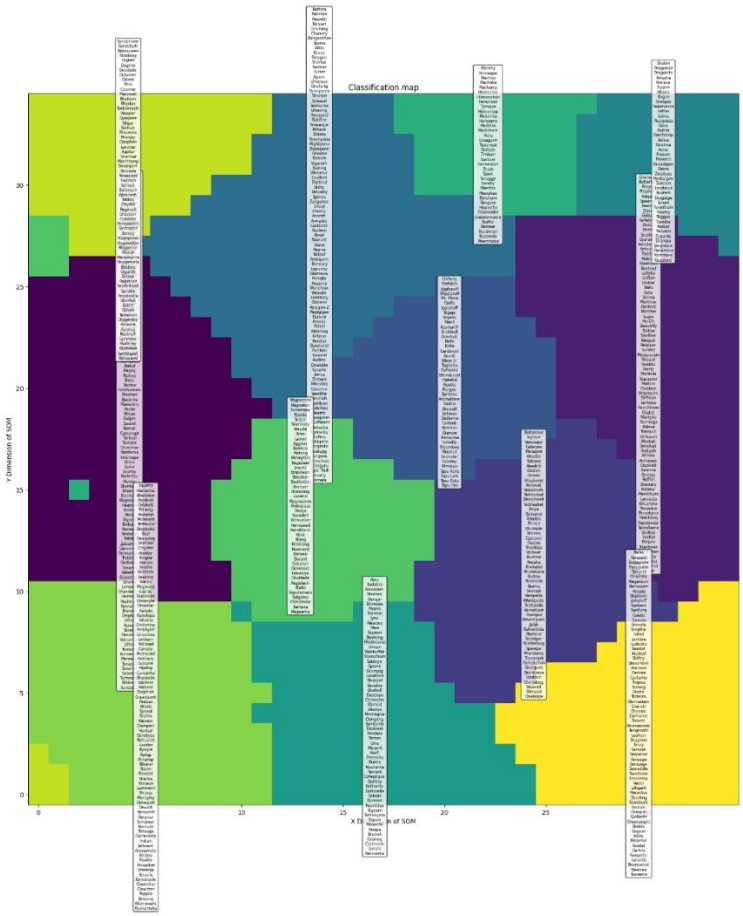

Figure 4-Classification Map clusters

This large classification map visually represents all the clusters, making it easy to see how the Pokemon are grouped. Each cluster includes a corresponding list of Pokemon assigned to it. The list associated with each cluster is positioned where it passes through the center of the cluster. This way helps improve readability, especially since some clusters contain many Pokemons, which can lead to overlapping. By aligning the lists centrally within their respective clusters, we ensure a clearer visualization of the classification.

In the next figure, we can see the list of Pokemons in each neuron, The spaces were there is nothing, it means that in those locations, there are no Pokemons.

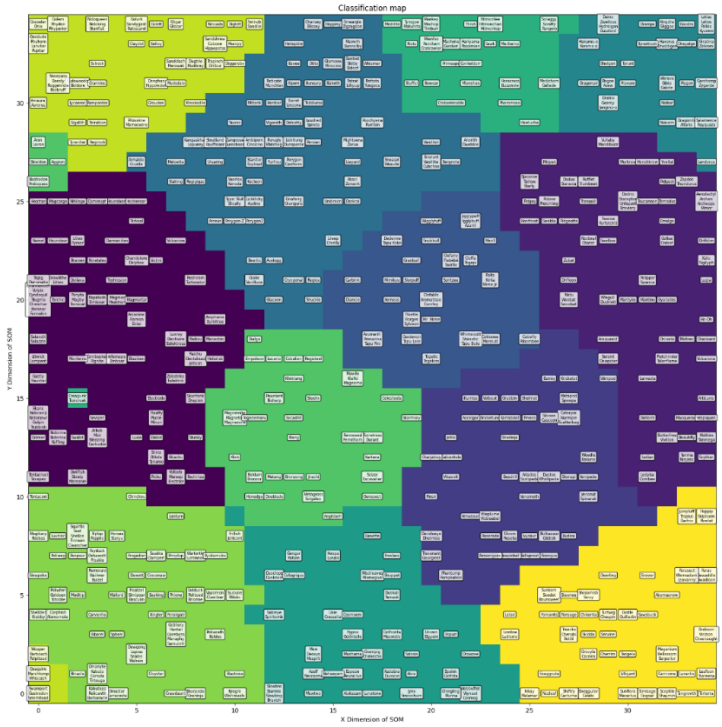

Figure 5-Pokemons per neuron

Activation map

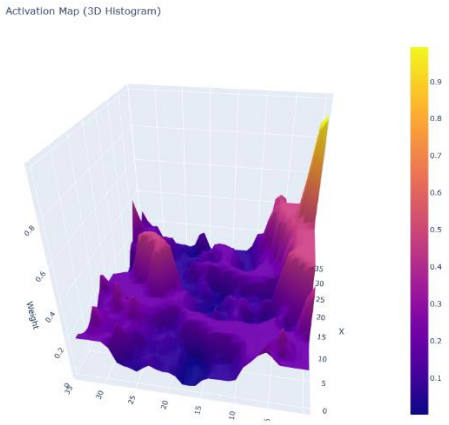

Figure 6-Activation Map

The 3D activation map provides a detailed visualization of how Pokemon are distributed across the Self-Organizing Map (SOM), with height representing the frequency of neuron activation. Taller peaks indicate regions where multiple Pokemon with similar attributes are mapped, forming well-defined clusters, while lower areas suggest neurons that are rarely activated, possibly highlighting outliers or unique Pokemon. The legend beside the diagram helps interpret activation intensities, where warmer colors (yellow in this case) signify highly active neurons and cooler colors (purple in this case) indicate lower activation. If certain Pokemon is mapped to low-activation regions, it may suggest they has distinct characteristics that need further examination.

Distance map

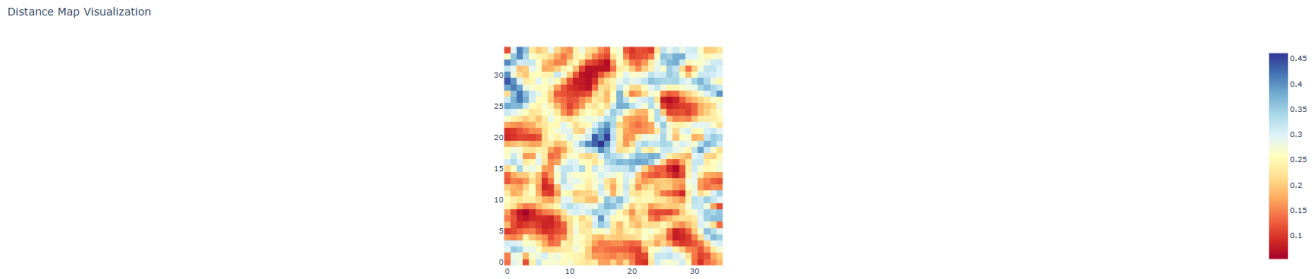

Figure 7-Distance Map

The distance map, also known as the U-Matrix (Unified Distance Matrix), provides insight into the separability of clusters within the Self-Organizing Map (SOM). It visualizes the distances between neighbouring neurons, where darker regions indicate larger distances, suggesting well-separated clusters, while lighter regions represent smaller distances, resulting in more closely related Pokemon. A high U-Matrix value between two neurons shows that they are far apart in the space (hinting towards a cluster boundary), while a low U-Matrix value means that the neurons are close in space (belonging to the same cluster). The Mexican Hat function helps the separation by emphasizing contrast, making the cluster boundaries in the U-Matrix more distinct.

Quantification error and topographic error

```
Hyperparameters: map_side=35, epochs=60000, initial_lr=0.05
Quantization Error: 0.2654
Topographic Error: 0.0879
```

Figure 8-Quantification Error and Topographic Error

After finding the adequate hyperparameters in our experimentation phase, we evaluated the model's performance using quantification error and topographic error to assess the accuracy and structure of the clustering. The quantification error measures how well the model represents the input data by calculating the average distance between each Pokemon and its best-matching unit (BMU). A lower quantification error indicates that the SOM effectively captures the underlying data distribution. On the other hand, the topographic error evaluates how well the SOM preserves the data topology by checking whether neighbouring data points in the input space remain nearby in the map. A high topographic error suggests that the model struggles to maintain the relationships between Pokemon attributes, while a low error confirms a well-structured representation.

2.4.2. Evaluation questions

Have the different examples been correctly classified? (From the Test dataset)

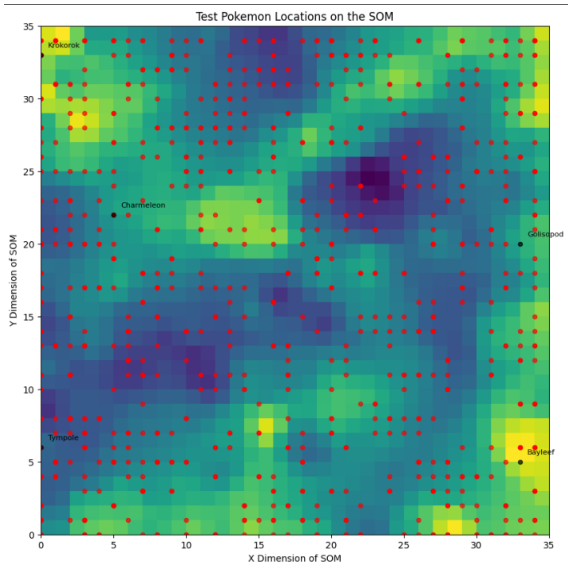

Figure 9-Pokemon Location in the SOM

```
Test Pokemon: Charmeleon: ['fire' nan]
Neighborhood Pokemon:
- Victini: ['psychic' 'fire']
- Chandelure: ['ghost' 'fire']
- Delphox: ['fire' 'psychic']
--------------------
Test Pokemon: Bayleef: ['grass' nan]
Neighborhood Pokemon:
- Parasect: ['bug' 'grass']
- Wormadam: ['bug' 'grass']
- Abomasnow: ['grass' 'ice']
- Leavanny: ['bug' 'grass']
--------------------
Test Pokemon: Tympole: ['water' nan]
Neighborhood Pokemon:
- Slowpoke: ['water' 'psychic']
--------------------
Test Pokemon: Krokorok: ['ground' 'dark']
Neighborhood Pokemon:
- Geodude: ['rock' 'ground']
- Graveler: ['rock' 'ground']
- Onix: ['rock' 'ground']
- Rhyhorn: ['ground' 'rock']
- Larvitar: ['rock' 'ground']
- Pupitar: ['rock' 'ground']
--------------------
Test Pokemon: Golisopod: ['bug' 'water']
Neighborhood Pokemon:
- Gyarados: ['water' 'flying']
--------------------
```

Figure 10-List of Neighbour Pokemon's to Test Pokemon's

After classifying the test set, we have seen that the Pokemon are correctly assigned to their respective neurons. Pokemons with similar attributes are mapped together, which indicates that the SOM has correctly learned and generalized the relationships in the data. To know the exact location of all the Pokemons in the map, we had to calculate the BMU for each of them.

Apart from generating a visual representation on the location of the test Pokemons in the SOM, we also printed a list of the name of each test pokemon and just under the Pokemons along with its neighbourhood Pokemon (within a range of 1). Next to each Pokemon, we printed out their type number to make it easier for us to know if they were correctly classified by their type.

As it can be seen in the results, the test Pokemons were placed in neighbourhoods that match their attributes, confirming that the SOM effectively grouped them based on their similarities.

In which cluster is Pikachu classified? Is is correct?

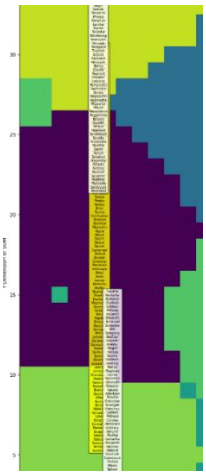

Figure 11-Pikachu's location in classification map

```
Cluster 0: Charmander, Ekans, Arbok, Pikachu, Raichu, Nidoran9, Nidorina, Nidoran♂, Nidorin
Cluster 1: Charizard, Butterfree, Pidgey, Pidgeotto, Pidgeot, Spearow, Fearow, Zubat, Golbat
Cluster 2: Bulbasaur, Ivysaur, Venusaur, Caterpie, Metapod, Weedle, Kakuna, Beedrill, Oddish
Cluster 3: Clefairy, Clefable, Jigglypuff, Wigglytuff, Mr. Mime, Cleffa, Igglybuff, Togepi,
Cluster 4: Rattata, Raticate, Meowth, Persian, Lickitung, Chansey, Kangaskhan, Tauros, Ditto
Cluster 5: Dratini, Dragonair, Dragonite, Kingdra, Vibrava, Flygon, Altaria, Bagon, Shelgon,
Cluster 6: Abra, Kadabra, Alakazam, Slowbro, Gengar, Drowzee, Hypno, Starmie, Jynx, Mewtwo,
Cluster 7: Mankey, Primeape, Machop, Machoke, Machamp, Hitmonlee, Hitmonchan, Heracross, Ty
Cluster 8: Magnemite, Magneton, Forretress, Steelix, Scizor, Skarmory, Mawile, Aron, Lairon,
Cluster 9: Squirtle, Wartortle, Blastoise, Psyduck, Golduck, Poliwag, Poliwhirl, Poliwrath,
Cluster 10: Sandshrew, Sandslash, Nidoqueen, Nidoking, Diglett, Dugtrio, Geodude, Graveler,
Cluster 11: Paras, Parasect, Exeggcute, Exeggutor, Tangela, Chikorita, Meganium, Bellossom,
```

Figure 12-Cluster where Pikachu is located

We haven't included the whole classification map due to that its size is very big, we have just cropped the map and highlighted the cluster list where Pikachu is classified.

Pikachu has been classified in a cluster with other Electric-type Pokemon. After checking the list, we can see that Pickachu is grouped with similar Pokemon, including Raichu which is its evolution.

In Figure 12, you can see the location of Pikachu as well as some other Pokemons in the same cluster.

With this result, we can see the effectiveness of our SOM training, confirming that the Pokemon's are classified based on their shared numerical characteristics.

Which are the features of the groups around the group in which Pikachu is classified?

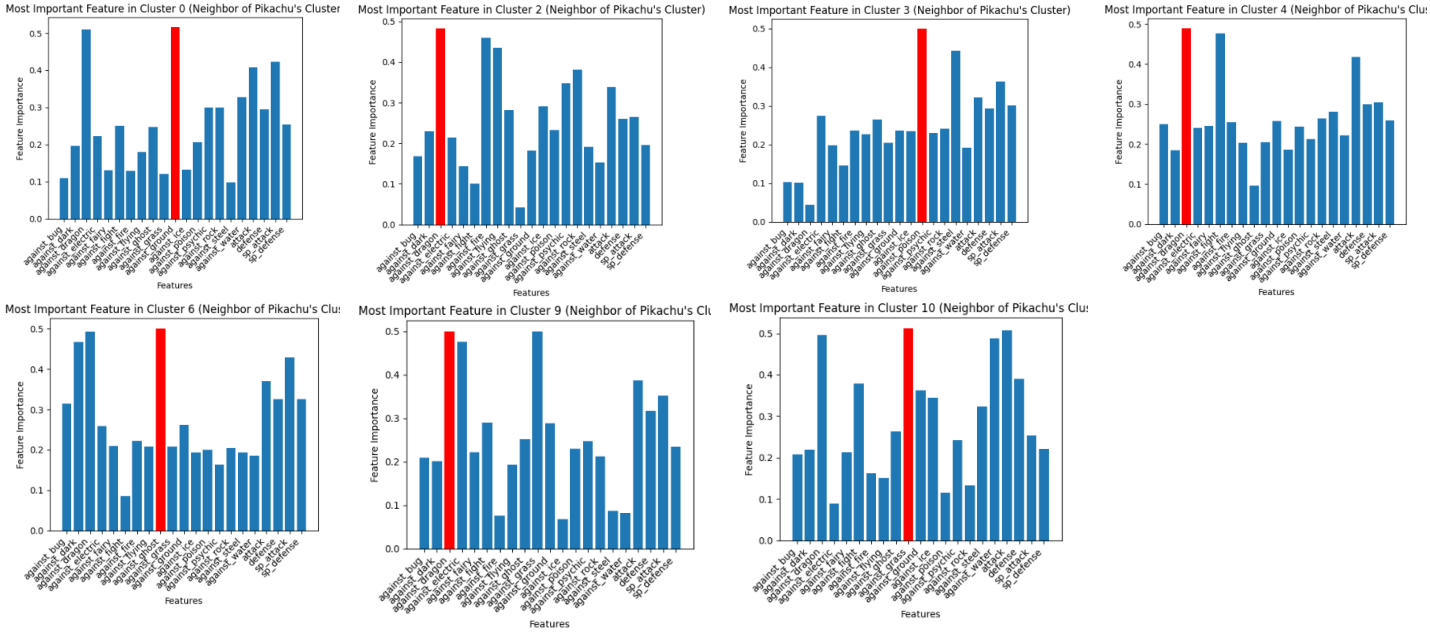

Figure 13-Most Important Features in the clusters next to Pikachu's Cluster

Since we removed the type attributes (type1 and type2) from our dataset and some others as explained above, in this visual representation, only the attributes used to train the SOM can be seen with a clear height difference of the attributes that played a stronger role in the classification. To analyze the features of the groups next to Pikachu's cluster, we examined the most influential attributes in each neighbouring cluster. The red highlighted bars in Figure 13 indicate the most significant attributes.

Are Articuno and Moltres grouped together? Why do you think this happens?

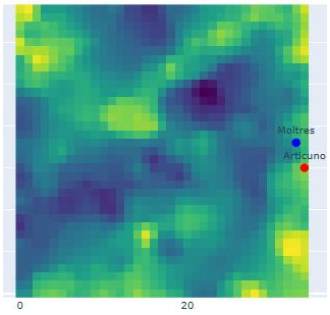

Figure 14-Location os Articuno and Moltres in SOM map

Figure 15-Names of pokemons in Articuno's and Moltres's cluster

We can see in the visual representation that both Pokemons are placed closed to each other in the SOM Map. To know their exact location of both Pokemons in the map, we calculated the BMU for each. Articuno and Moltres are next to each other, which makes sense considering their similarities.

Answering the question on why this must be, it has an easy explanation. Both Pokemon's share the Flying type, which means they have similar resistance and weaknesses. Additionally, their base stats are highly similar, particularly in special attack and special defense, making them natural candidates for the same cluster. Since they are part of the Legendary Birds, their overall structure follows a similar pattern, which explains why they are placed in the same area on the map.

What significance would you attribute to the cluster in which Slowbro is placed?

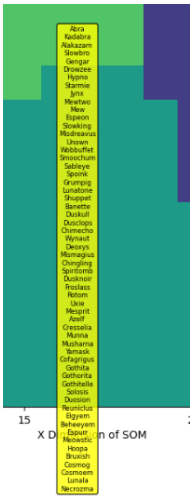

Figure 16-Slowbro's Cluster

The cluster where Slowbro is located is important because it groups Pokemon that share key similarities, such as Psychic typing, high defense and special attack. Many of the Pokemon in this cluster like Alakazam, Espeon, and Mewtwo, are primarily Psychic-type, suggesting that Slowbro's secondary typing influenced its classification. Also, since this group includes several defensive Pokemon, it shows that the SOM considered both type and battle style when forming the clusters.

3. Conclusion

To conclude, in this practical work we have learned and achieved how to do a good implementation and analysis of a Self-Organizing Map (SOM) for classifying Pokemons based on their different attributes.

Initially, we faced challenges in defining the most relevant features and tuning the SOM's hyperparameters. However, through iterative experimentation, we refined our approach and successfully developed a model that provides meaningful Pokemon groupings.

After carefully examining, transforming, and normalizing the dataset, we removed non-essential attributes while keeping those that contributed the most to classification. During the implementation phase, we tested different configurations for the map size, learning rate, and neighborhood size to train multiple models. Choosing the optimal model was not just about finding the one with the lowest error but also ensuring clear and meaningful clusters. We supported our decision using classification maps, activation maps, and error measurements such as quantization and topographic errors.

After evaluating various configurations, we determined that the best performing model had a map side equal to 35, epochs equal to 60000, and initial learning rate equal to 0.05, and achieving a good SOM's map with clear clusters, we have been able to answer the different questions required.

4. Bibliography

[1] J. Brownlee, "A Gentle Introduction to Distance Measures in Machine Learning," *Machine Learning Mastery*, 2019. [Online]. Available: https://machinelearningmastery.com/distance-measures-for-machine-learning/

[2] J. Dancker, "A Brief Introduction to Distance Measures," *Medium*, 2019. [Online]. Available: https://medium.com/@jodancker/a-brief-introduction-to-distance-measures-ac89cbd2298

[3] J. Sirosh and R. Miikkulainen, "How Lateral Interaction Develops in a Self-Organizing Feature Map," *Proceedings of the IEEE International Conference on Neural Networks*, 1993. [Online]. Available: https://nn.cs.utexas.edu/downloads/papers/sirosh.lateral-interaction.pdf

[4] "Self-Organizing Maps (Kohonen Maps) Competitive learning," Philadelphia University. [Online]. Available: https://www.philadelphia.edu.jo/academics/qhamarsheh/uploads/Lecture%2015_Self-Organizing%20Maps%20%28Kohonen%20Maps%29.pdf

[5] "Self-organizing Maps," Harvey Mudd College. [Online]. Available: https://www.cs.hmc.edu/~kpang/nn/som.html

[6] "Performance evaluation of the self-organizing map for feature extraction," *Journal of Geophysical Research: Oceans*, vol. 110, no. C5, 2005. [Online]. Available: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2005JC003117