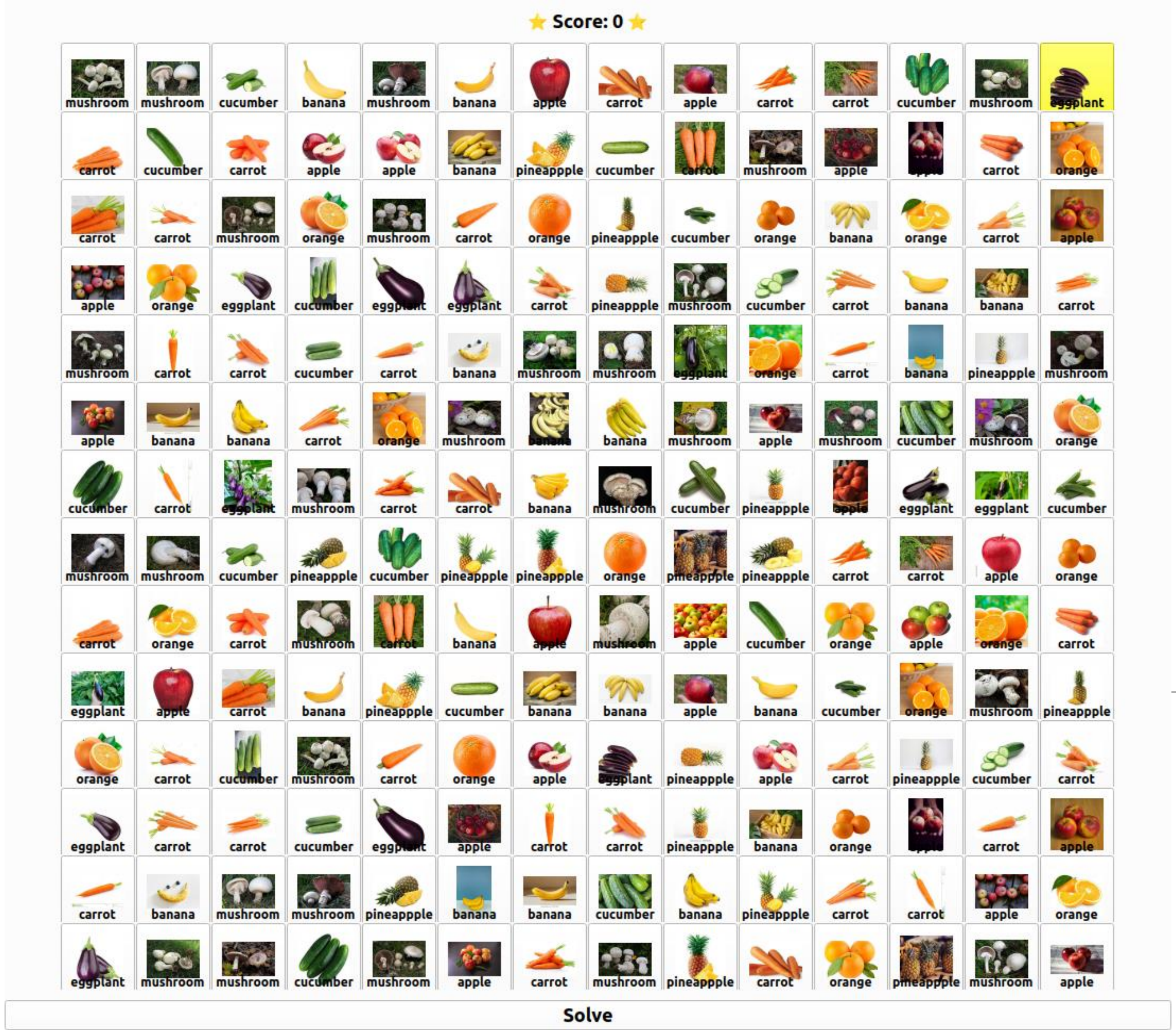


⭐ Score: 0 ⭐

mushroom	mushroom	cucumber	banana	mushroom	banana	apple	carrot	apple	carrot	carrot	cucumber	mushroom	eggplant
carrot	cucumber	carrot	apple	apple	banana	pineappple	cucumber	carrot	mushroom	apple	apple	carrot	orange
carrot	carrot	mushroom	orange	mushroom	carrot	orange	pineappple	cucumber	orange	banana	orange	carrot	apple
apple	orange	eggplant	cucumber	eggplant	eggplant	carrot	pineappple	mushroom	cucumber	carrot	banana	banana	carrot
mushroom	carrot	carrot	cucumber	carrot	banana	mushroom	mushroom	eggplant	orange	carrot	banana	pineappple	mushroom
apple	banana	banana	carrot	orange	mushroom	banana	banana	mushroom	apple	mushroom	cucumber	mushroom	orange
cucumber	carrot	eggplant	mushroom	carrot	carrot	banana	mushroom	cucumber	pineappple	apple	eggplant	eggplant	cucumber
mushroom	mushroom	cucumber	pineappple	cucumber	pineappple	pineappple	orange	pineappple	pineappple	carrot	carrot	apple	orange
carrot	orange	carrot	mushroom	carrot	banana	apple	mushroom	apple	cucumber	orange	apple	orange	carrot
eggplant	apple	carrot	banana	pineappple	cucumber	banana	banana	apple	banana	cucumber	orange	mushroom	pineappple
orange	carrot	cucumber	mushroom	carrot	orange	apple	eggplant	pineappple	apple	carrot	pineappple	cucumber	carrot
eggplant	carrot	carrot	cucumber	eggplant	apple	carrot	carrot	pineappple	banana	orange	apple	carrot	apple
carrot	banana	mushroom	mushroom	pineappple	banana	banana	cucumber	banana	pineappple	carrot	carrot	apple	orange
eggplant	mushroom	mushroom	cucumber	mushroom	apple	carrot	mushroom	pineappple	carrot	orange	pineappple	mushroom	apple

ME536 Semester Project:

Fresh Crush

Zeynep Neslişah Yılmaz

Fresh Crush Game

Candy crush like game with fruits and vegetables.

Problems:

- The game should learn the classification of the fruits and vegetables → ANN
- What if the game is caught off guard? Novalties ? → ANN & DATA CLUSTRING
- If you've learned to predict, step for crush them → SEARCH

Prototypical Networks

The model learns to **map each class to a prototype** (a representative feature vector).

Efficient Representation – Uses prototypes instead of storing all instances, reducing memory and computation.

Better Generalization – Captures category essence, improving classification of unseen data.

Fast & Scalable Learning – Learns efficiently, ideal for few-shot learning.

Adaptive to Concept Drift – Easily updates to reflect new data patterns.

Let's start to procedure

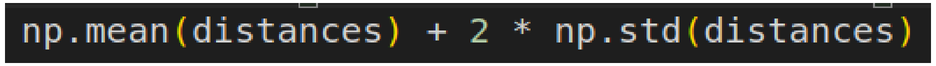
Construct the embeddings for supervised class

- ResNet18 is used as a **feature extractor** with removing the classification layer**.**
- With validation data the **model is fine-tuned (**Backpropagation & optimization steps)
- **Cosine similarity** is measured between embeddings.
- **Prototypes, trained embeddings,** and **the fine-tuned model** are **stored**

To enhance the quality of feature vectors obtained from the prototypical neural network, background removal preprocessing is performed using the rembg Python library, which removes backgrounds using deep learning models, primarily U^2-Net.

Discover and Classify Novelty

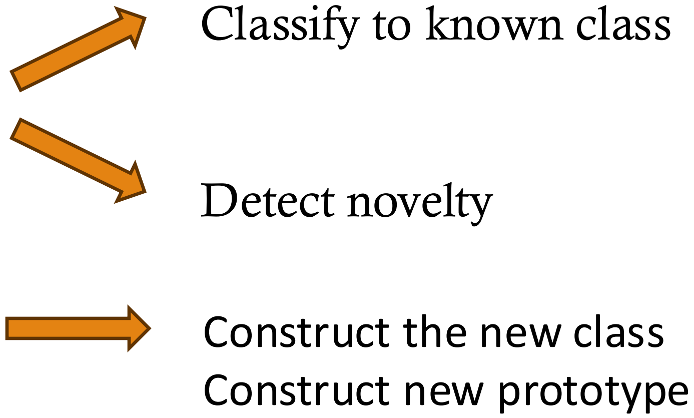
- The **novelty threshold** is defined by examining the cosine similarity between training embeddings and their corresponding prototypes.



```
np.mean(distances) + 2 * np.std(distances)
```

- Compare the embedding of a new instance with the threshold:

- If the same novelty comes 3 times



Classify to known class

Detect novelty

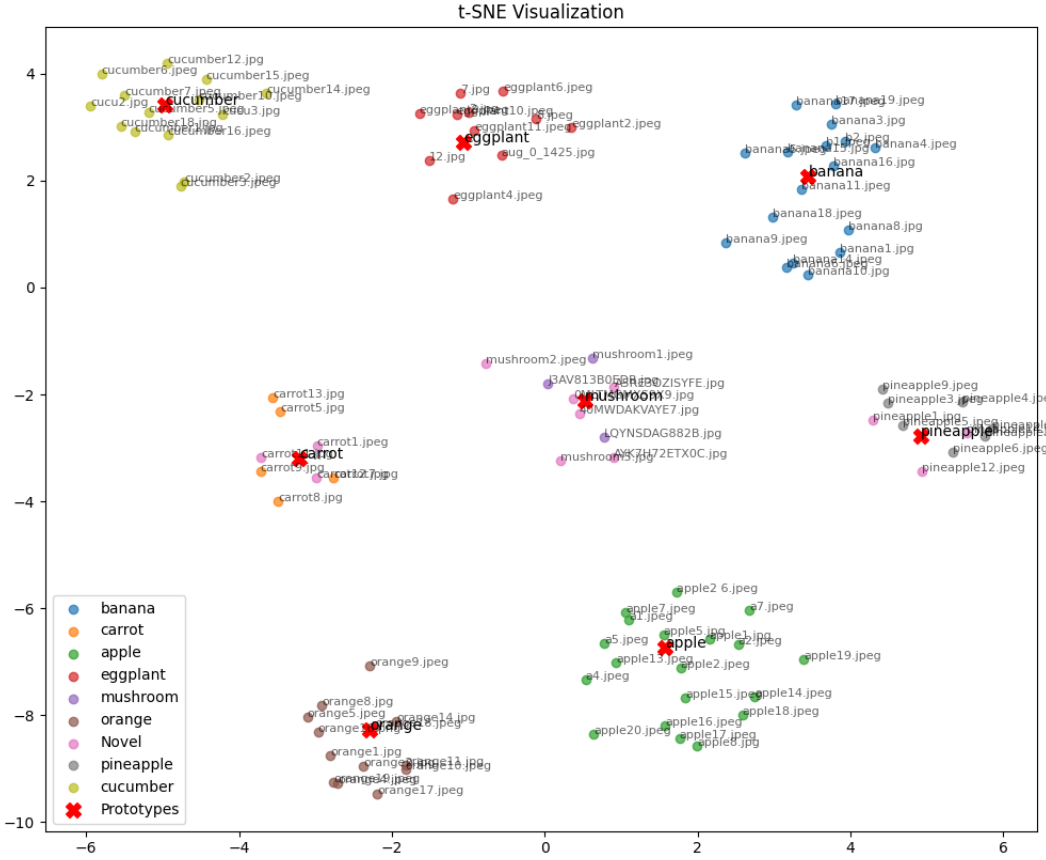
Construct the new class
Construct new prototype

Dynamic Clustering

- **DBSCAN**, an unsupervised clustering algorithm that groups data points based on density, is used for clustering the novelties.

Why Use DBSCAN for Novelty Detection?

- It **automatically detects clusters** without needing a predefined number (k in k-means).
- It **ignores noise**, meaning it can filter out outliers.
- It works well for **non-uniform cluster shapes**.



Fresh Crush Game 🍎🍌🍊🍆🥒🍍🥕🍄

Game with 2 levels: **Level 1 uses supervised learning, while Level 2 involves novelty detection.**

Based on the player's level choice, the corresponding dataset is assigned.

The images are randomly chosen.

Fruit classification is performed, and the game table is constructed.

The next step is the search algorithm.

Search Algorithm

Exhaustive search with greedy selection is used to find the **best move** by evaulating **possible swaps** and their **cascading effect**s. Since all possible moves does not take much time in this scenario.

Score calculation: The score is determined by (length of the crush * multipler effect), where multiplier:

- 1 for known classes.
- For novelties, it is defined by the user.

If cascading occurs frequently, the search may take longer. A heuristic approach can be used:

- Using the 2-3 steps of cascade score as reference.
- Prioritizing the match if it have two same class fruits near it

Further improvements

- Deeper networks for feature extractor can be used.
- Prototypes can be calculated with larger dataset.
- Instead of static prototypes, learnable prototypes can be used
- Attention mechanism can be used to refine prototype representation
- Gradient based meta learning can be integrated (MAML)
- Search method can be improved for larger datasets.
- The game logic can be enhanced with additional features.

Let's play the Fresh Crush !!