Python-Louvain is a popular library for detecting communities in graphs using the **Louvain method**, a greedy optimization algorithm that maximizes modularity. This guide covers installation, basic usage, and practical examples.

---

1. Installation

Install via pip

```bash
pip install python-louvain
```

(Alternatively called `community` in some versions.)

Verify Installation

```python
import community as community_louvain
print(community_louvain.__version__)
```

Dependencies

- Requires `networkx` (`pip install networkx`)
- Optional: `matplotlib` for visualization (`pip install matplotlib`)

---

2. Basic Usage

Import Required Libraries

```python
import networkx as nx
import community as community_louvain  # Main Louvain library
import matplotlib.pyplot as plt  # For visualization (optional)
```

Create or Load a Graph

```python
# Example: Create a random graph
G = nx.erdos_renyi_graph(100, 0.1)  # 100 nodes, 10% edge probability

# Or load a real-world dataset
# G = nx.read_edgelist("social_network.txt")
```

3. Detect Communities

Run Louvain Algorithm

```python
partition = community_louvain.best_partition(G)
print(partition)  # Returns {node1: community_id1, node2: community_id2, ...}
```

Get Modularity Score

Modularity measures the strength of community structure (higher = better clustering).

```python
modularity = community_louvain.modularity(partition, G)
print("Modularity:", modularity)  # Typically between -0.5 and 1
```

Count Communities

```python
num_communities = max(partition.values()) + 1
print("Number of communities:", num_communities)
```

4. Visualize Communities

Color Nodes by Community

```python
```

```python
# Assign colors based on community ID
cmap = plt.cm.get_cmap("viridis", max(partition.values()) + 1)
nx.draw_spring(
    G,
    node_color=[partition[i] for i in G.nodes()],
    cmap=cmap,
    with_labels=False,
    node_size=50,
)
plt.title("Louvain Community Detection")
plt.show()
```

Alternative Visualization (Grouped Layout)

```python
pos = nx.spring_layout(G)
nx.draw(G, pos, node_color=list(partition.values()), cmap=plt.cm.tab20)
plt.show()
```

5. Advanced Usage

Resolution Parameter (γ)

Adjusts community size sensitivity (higher γ → smaller communities).

```python
partition = community_louvain.best_partition(G, resolution=1.5)  # Default=1.0
```

Benchmarking & Comparing Partitions

```python
from networkx.algorithms.community import modularity

# Compare two partitions
partition2 = some_other_algorithm(G)
mod1 = modularity(G, [set(n for n, c in partition.items() if c == i) for i in set(partition.values())])
mod2 = modularity(G, [set(n for n, c in partition2.items() if c == i) for i in set(partition
2.values())])
print("Louvain modularity:", mod1, "Other method:", mod2)
```

6. Exporting Results

Save Communities to CSV

python Copy Download

```python
import pandas as pd
pd.DataFrame({"Node": partition.keys(), "Community": partition.values()}).to_csv("communities.csv")
```

Convert to NetworkX Communities Format

python Copy Download

```python
communities = {}
for node, comm_id in partition.items():
    if comm_id not in communities:
        communities[comm_id] = []
    communities[comm_id].append(node)
print("Communities:", communities)
```

7. Performance Tips

- **Large Graphs?** Use `resolution < 1.0` for fewer, larger communities.
- **Directed Graphs?** Convert to undirected first (`G = G.to_undirected()`).
- **Weighted Graphs?** Louvain automatically handles edge weights (`G.add_edge(u, v, weight=5)`).

8. Troubleshooting

Issue	Solution
`ModuleNotFoundError: No module named 'community'`	Install with `pip install python-louvain`
Low modularity (<0.3)	Try adjusting `resolution` or preprocess the graph
Slow on large networks	Use a sparse graph or reduce `resolution`

9. Alternatives

- **Leiden Algorithm** (faster, more stable): `pip install leidenalg`
- **Girvan-Newman** (hierarchical clustering): `nx.algorithms.community.girvan_newman(G)`

Conclusion

Python-Louvain provides:
✅ Fast community detection
✅ Modularity optimization
✅ Easy integration with `networkx`

For more details, see the GitHub repository.