Topluluk Tespiti: Klasik Algoritmalar ve Çizge Sinir Ağları Karşılaştırması

Emre YILDIZ

Ege Üniversitesi
Bilgisayar Mühendisliği Bölümü

June 19, 2025

Proje Hakkında

- **Ders:** Cebirsel Çizge Algoritmaları (Yüksek Lisans)
- **Amaç:** Çizge Teorisi'nin teorik temellerini modern Makine Öğrenmesi uygulamaları ile birleştiren kapsamlı bir topluluk tespiti (community detection) çalışmasıdır.
- **Yöntemler:** Neo4j ve PyTorch kullanarak klasik algoritmaları (Louvain, LPA) Graph Neural Networks (GCN, GraphSAGE, GAT) ile karşılaştırmak.
- **Veriseti:** Cora Citation Network.

Kullanılan Kütüphane ve Araçlar

- **Neo4j (GDS):** Yüksek performanslı çizge veritabanı. Louvain ve Label Propagation gibi klasik topluluk tespit algoritmalarını çalıştırmak için kullanıldı.
- **PyTorch (PyG):** Derin öğrenme ve özellikle Çizge Sinir Ağları (GNN) modelleri oluşturmak ve eğitmek için kullanılan esnek bir kütüphane.
- **NetworkX:** Çizge oluşturma, manipülasyonu ve temel çizge metriklerinin hesaplanması için kullanıldı.
- **Scikit-learn:** GNN modellerinin performansını değerlendirmek için Adjusted Rand Index (ARI) ve Normalized Mutual Information (NMI) gibi metriklerin hesaplanmasında kullanıldı.
- **Seaborn & Matplotlib:** Veri ve sonuçların görselleştirilmesi, karşılaştırma grafikleri oluşturulması için kullanıldı.

Neo4j ve Graph Data Science (GDS) Rolü

Projedeki İşlevi ve İş Akışı

- **Veri Depolama ve Modelleme:** Cora veriseti, Neo4j'de çizge yapısına en uygun şekilde depolanmıştır:
  - Düğümler: `:Paper` etiketi ile makaleler.
  - İlişkiler: `[:CITES]` ilişkisi ile makaleler arası alıntılar.
- **In-Memory Çizge Projeksiyonu:** Analizden önce, GDS kütüphanesi ile `'cora-graph'` adında bir "in-memory" yansıtma oluşturulur. Bu, algoritmaların disk I/O olmadan çok yüksek hızda çalışmasını sağlar.
- **Klasik Algoritmaların Çalıştırılması:**
  - `gds.louvain.write(...)` ve `gds.labelPropagation.write(...)` fonksiyonları ile topluluk tespiti algoritmaları çalıştırılmıştır.
  - `writeProperty` parametresi sayesinde, her bir algoritmanın bulduğu topluluk kimliği (`louvainCommunityId`, `lpaCommunityId`) doğrudan ilgili düğümlerin bir özelliği olarak veritabanına geri yazılmıştır.
- **Entegrasyon Noktası:** Neo4j, hem klasik algoritmaların analiz merkezi hem de PyTorch ile eğitilen GNN modelleri için bir veri kaynağı görevi görmüştür.

Louvain Algoritması

Çalışma Prensibi:

- Hiyerarşik bir kümeleme algoritmasıdır ve amacı ağdaki modülerlik skorunu maksimize etmektir.
- **1. Aşama (Modülerlik Optimizasyonu):** Her düğüm, komşu düğümlerin bulunduğu toplulukları tek tek dener. Düğüm, modülerlikte en büyük artışı sağlayan topluluğa taşınır. Bu işlem, hiçbir düğüm hareketi modülerliği artırmayana kadar tekrarlanır.
- **2. Aşama (Topluluk Birleştirme):** İlk aşamada

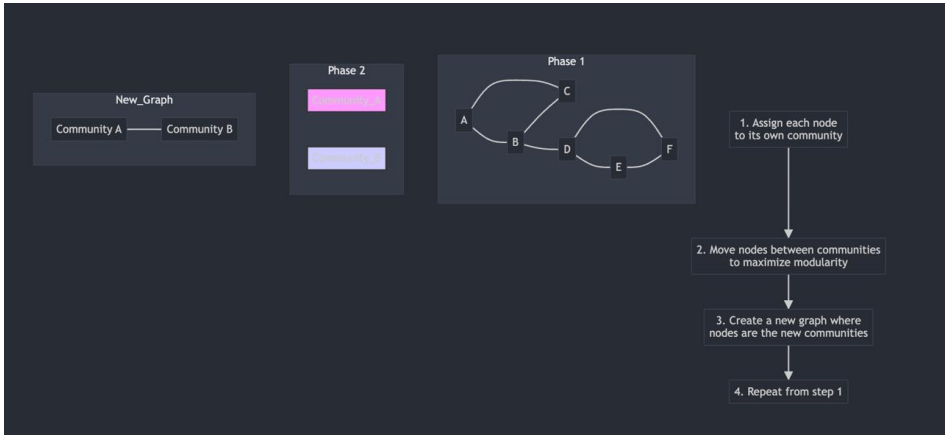


Figure: Louvain algoritmasının iki aşamalı yapısı.

Label Propagation (LPA)

Çalışma Prensibi:

- Yarı denetimli bir algoritmadır ve düğümlerin etiketlerini komşularına yayarak toplulukları bulur.
- **Başlangıç:** Her düğüme benzersiz bir etiket atanır.
- **İterasyon:** Her adımda, her düğüm komşularının etiketlerine bakar ve komşuları arasında en sık görülen (majoritary) etiketi kendi etiketi olarak günceller.
- **Duruş Kriteri:** Bu işlem, hiçbir düğümün etiketini değiştirmediği bir denge durumuna ulaşılana kadar tekrarlanır.

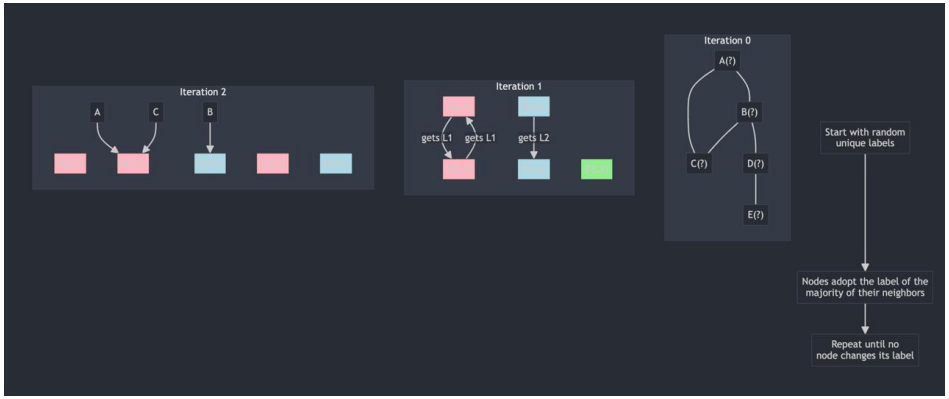


Figure: Etiketlerin komşular arasında yayılması.

Graph Convolutional Network (GCN)

Çalışma Prensibi:

- Çizge evrişim (convolution) operatörünü kullanarak düğüm temsillerini (embeddings) öğrenir.
- Her katmanda, bir düğüm kendi komşu düğümlerinin özellik vektörlerini toplar (aggregation) ve ortalamasını alır.
- Bu toplanan bilgi, düğümün kendi özellik vektörü ile birleştirilir.
- Sonuç, bir aktivasyon fonksiyonundan (örneğin ReLU) geçirilerek düğümün yeni katmandaki temsili oluşturulur.

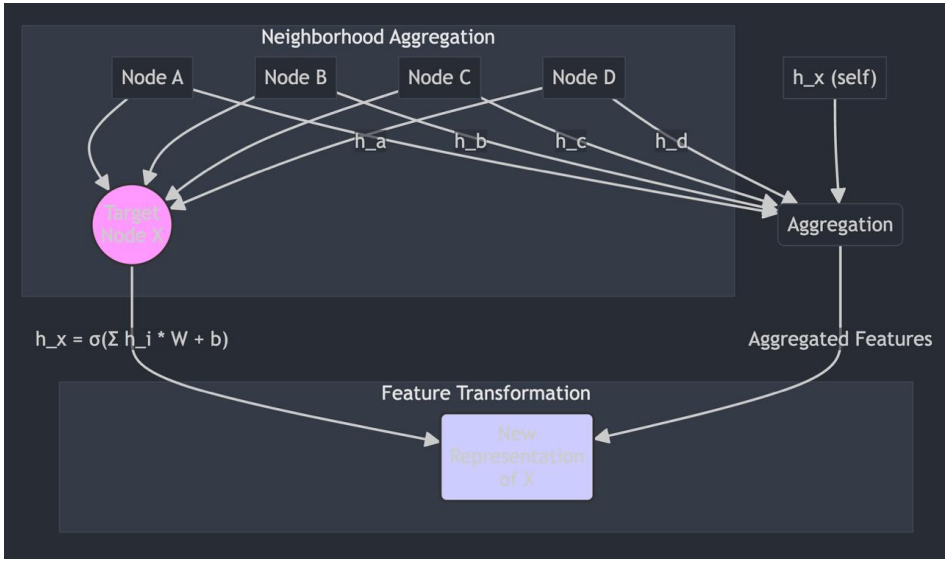
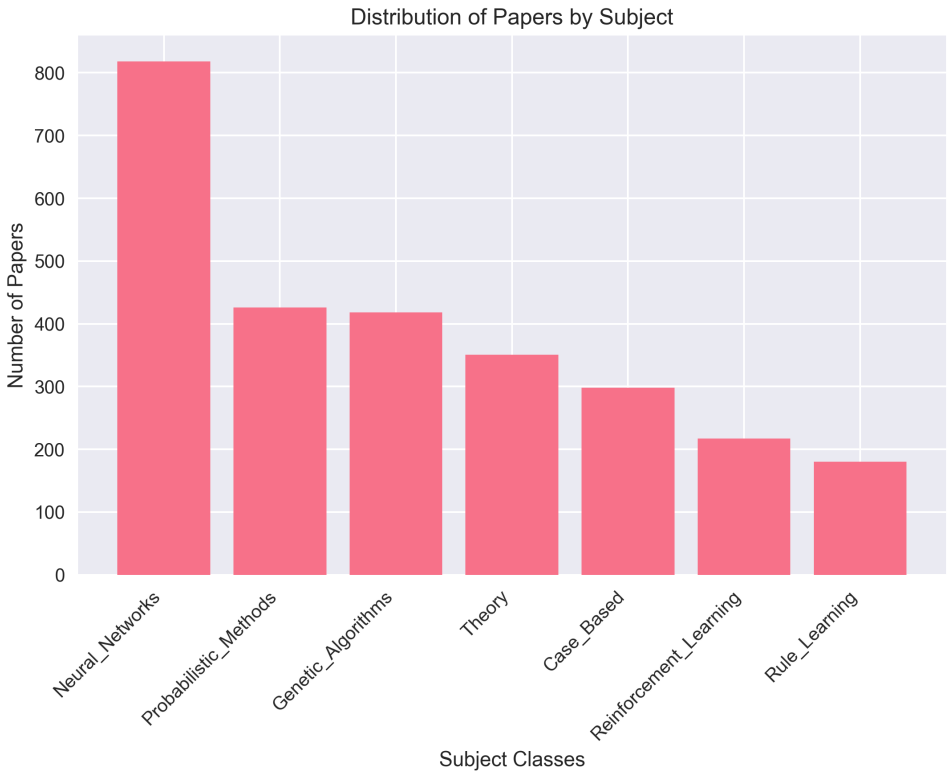


Figure: Komşuluk bilgilerinin toplanması ve güncellenmesi.

Sonuçlar: Veriseti İstatistikleri



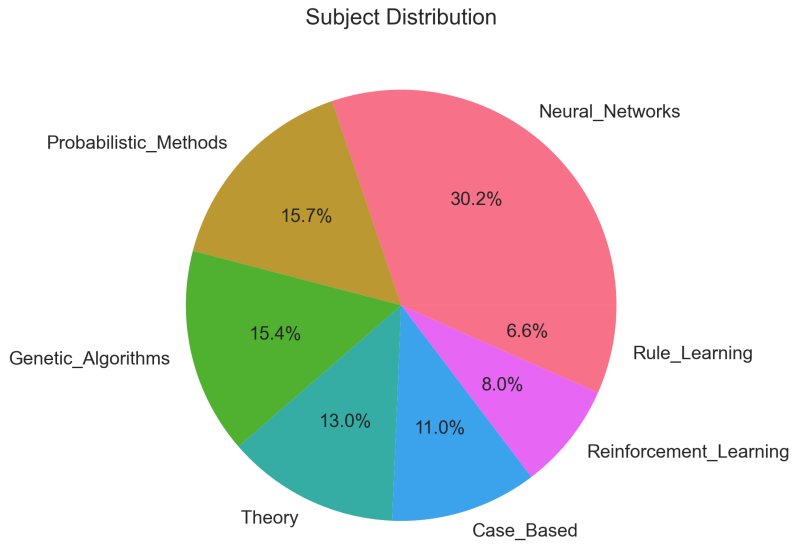
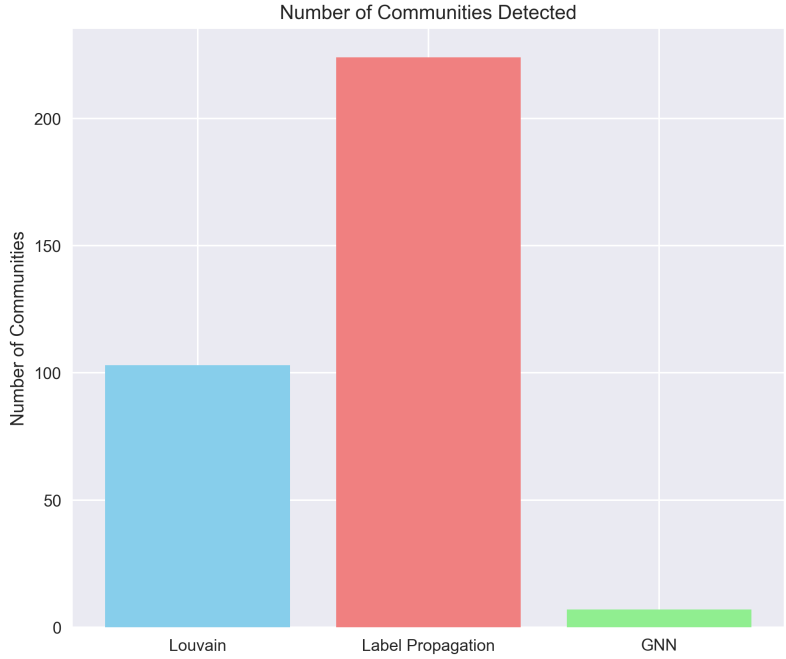
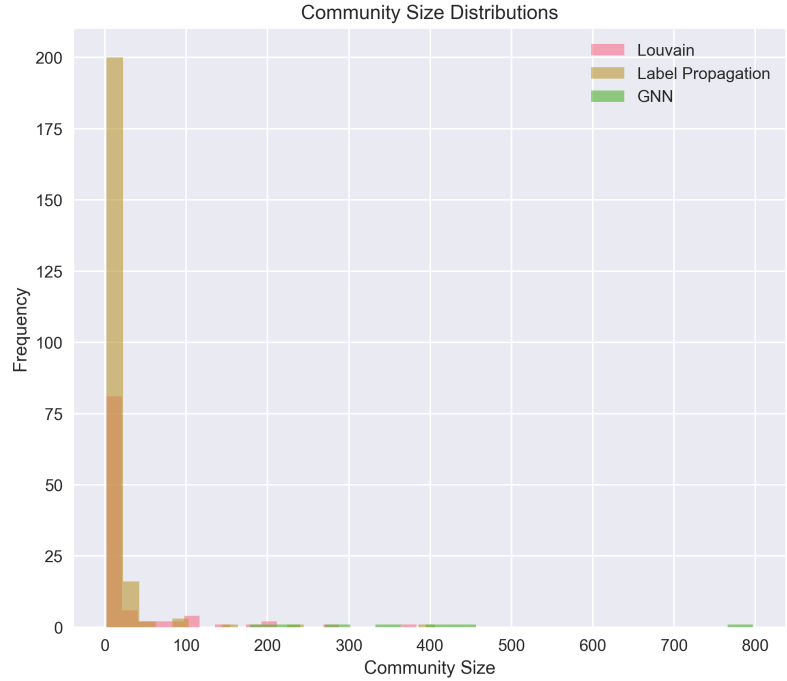
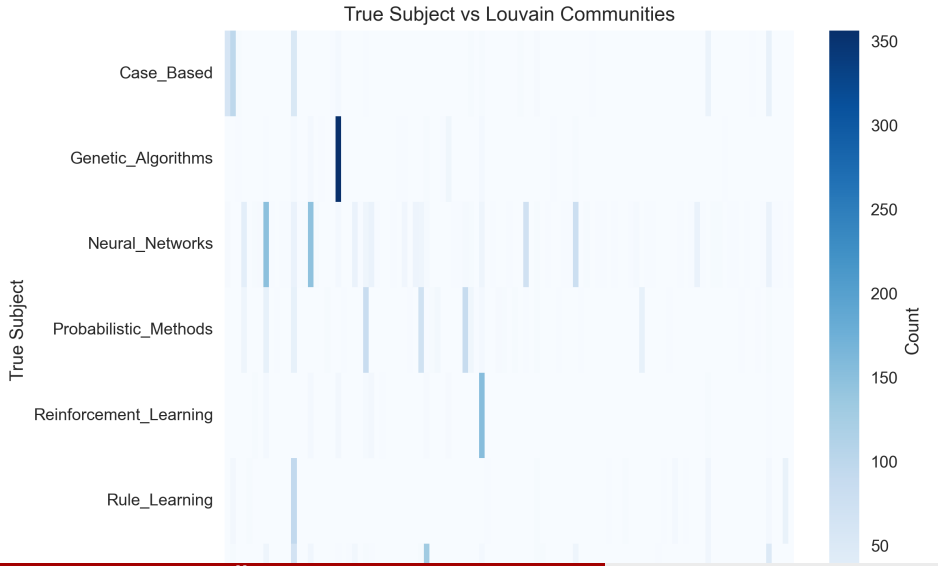


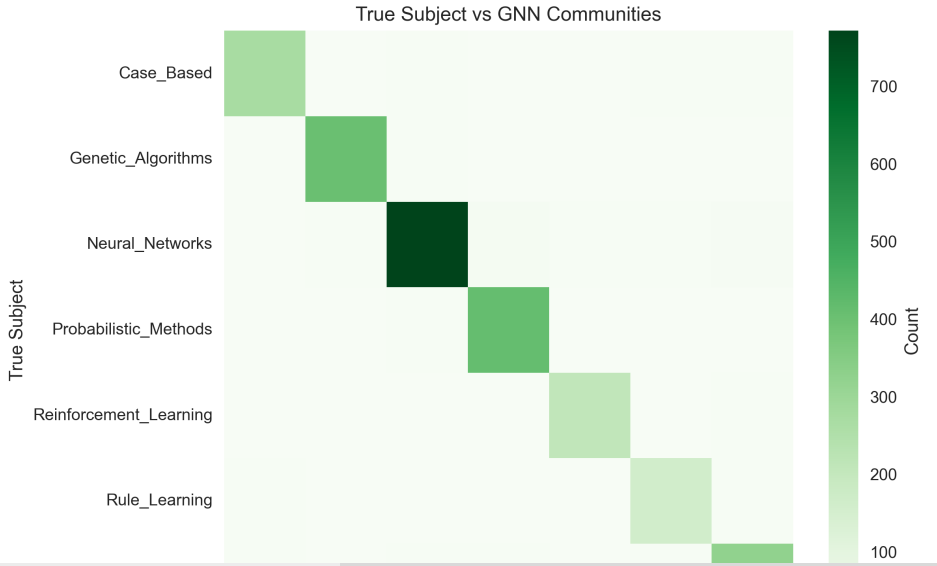
Figure: Cora verisetinin temel istatistikleri ve derece dağılımı.

Sonuçlar: Topluluk Karşılaştırması









Sonuçlar: Değerlendirme Metrikleri

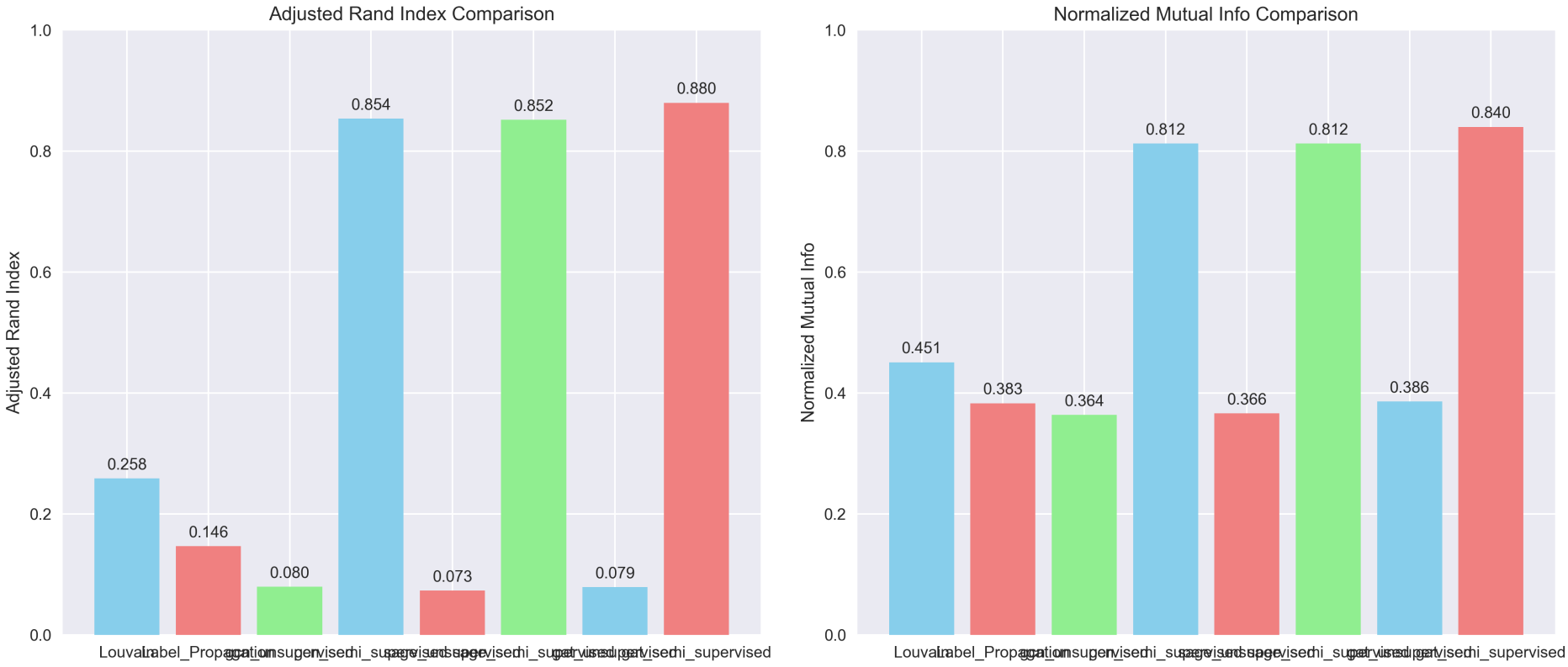


Figure: Algoritmaların ARI, NMI ve Modülerlik metriklerine göre performans karşılaştırması.

Sonuçlar: t-SNE ile Düğüm Gösterimi (Figure 1)

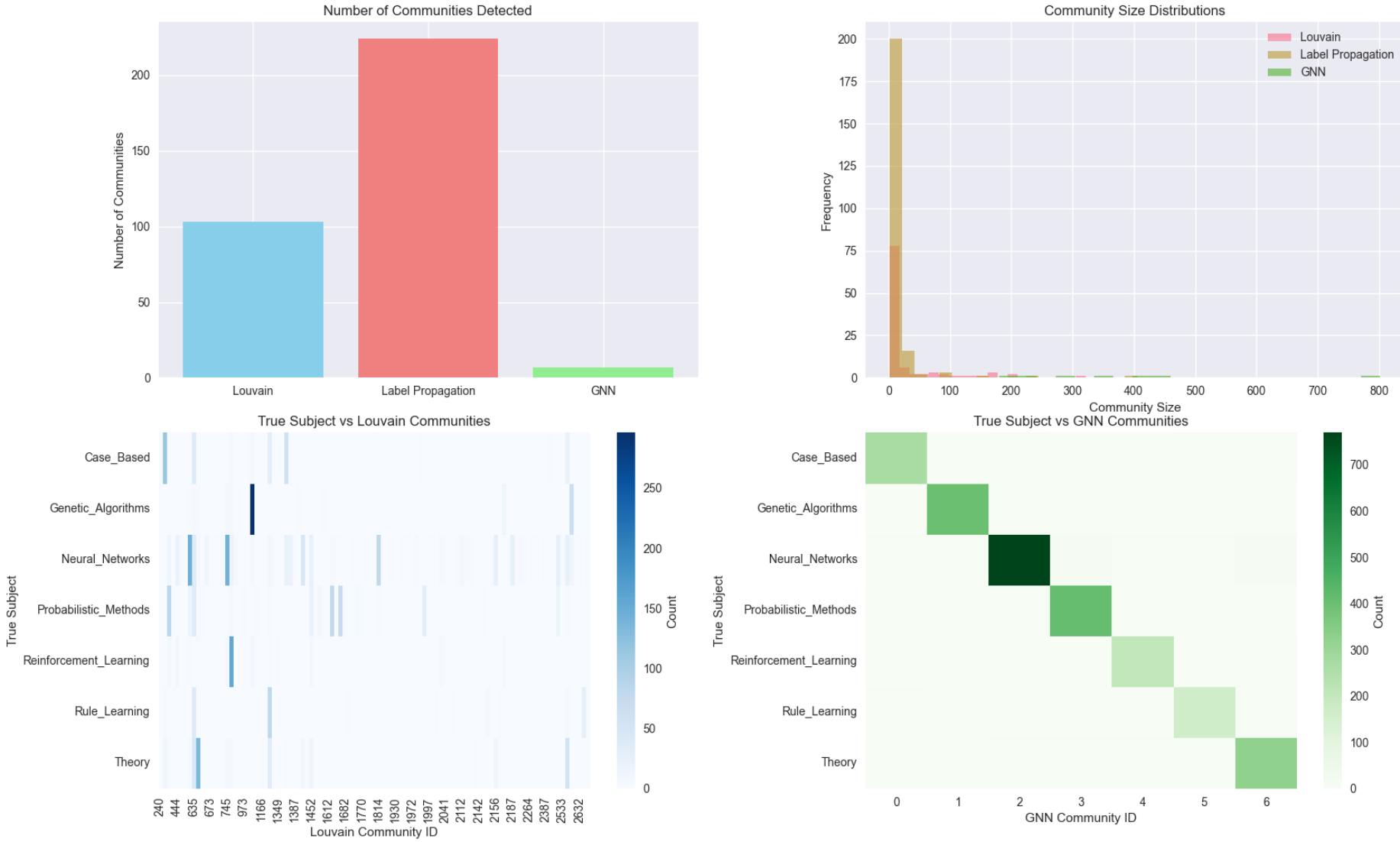


Figure: GCN modeli tarafından öğrenilen düğüm gömülmelerinin (embeddings) t-SNE ile 2 boyuta indirgenmiş hali

Sonuçlar: t-SNE ile Düğüm Gösterimi (Figure 2)

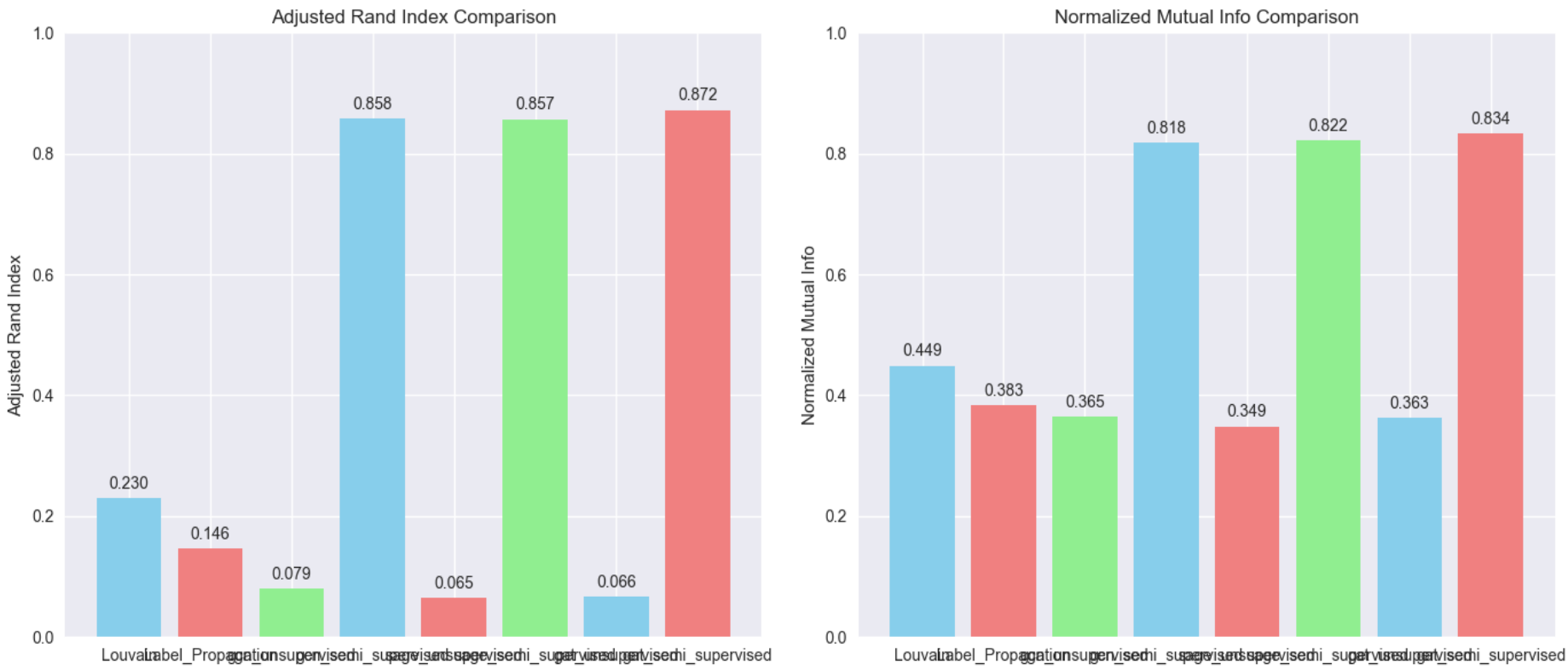


Figure: GraphSAGE modeli tarafından öğrenilen düğüm gömülmelerinin (embeddings) t-SNE ile 2 boyuta indirgenmiş hali.

Teşekkürler