

Topluluk Tespiti: Klasik Algoritmalar ve Çizge Sinir Ağları Karşılaştırması

Emre YILDIZ

Ege Üniversitesi
Bilgisayar Mühendisliği Bölümü

June 19, 2025

- **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:
 - Dğümler: :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 Prensipleri:

- 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

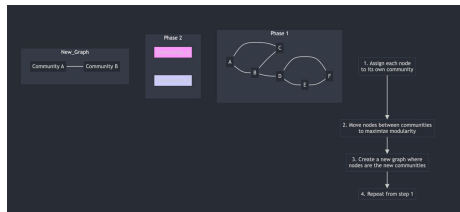


Figure: Louvain algoritmasının iki aşamalı yapısı.

Label Propagation (LPA)

Çalışma Prensipleri:

- 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 değiştirmedeği bir denge durumuna ulaşılan kadar tekrarlanır.

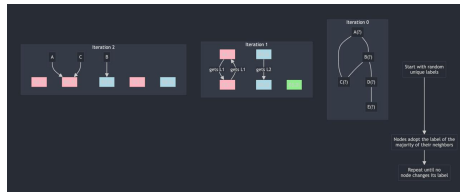


Figure: Etiketlerin komşular arasında yayılması.

Graph Convolutional Network (GCN)

Çalışma Prensipleri:

- 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.

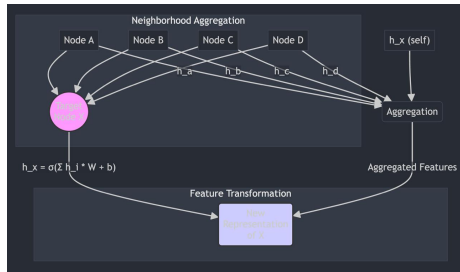


Figure: Komşuluk bilgilerinin toplanması ve güncellenmesi.

Sonuçlar: Veriseti İstatistikleri

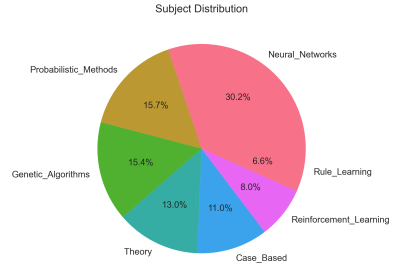
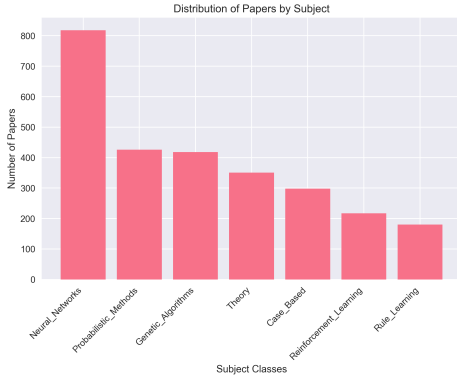
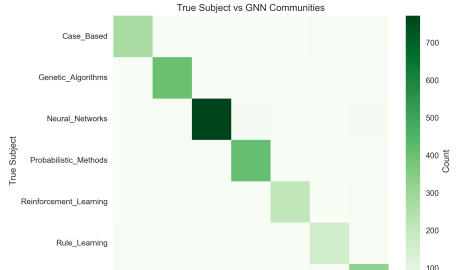
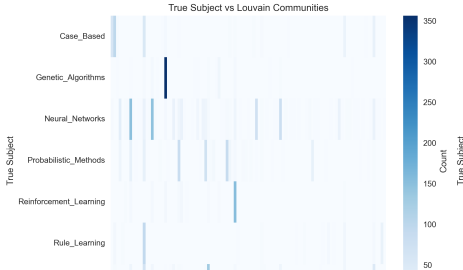
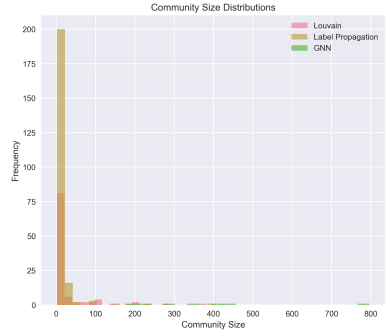
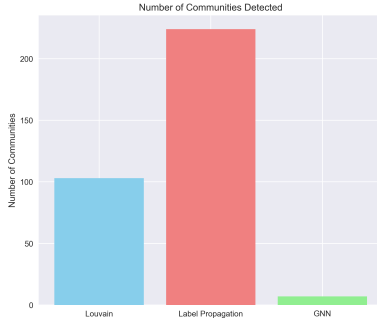


Figure: Cora verisetinin temel istatistikleri ve derece dağılımı.

Sonuçlar: Topluluk Karşılaştırması



Sonuçlar: Değerlendirme Metrikleri

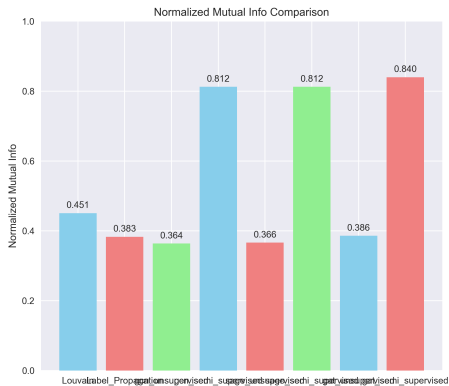
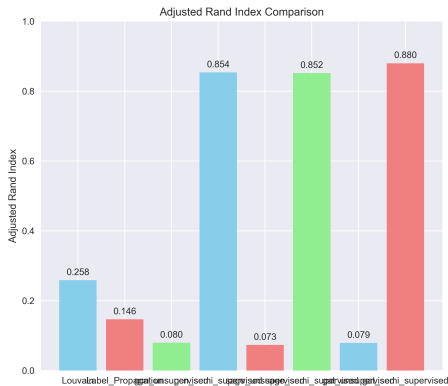


Figure: Algoritmaların ARI, NMI ve Modülerlik metriklerine göre performans karşılaştırması.

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

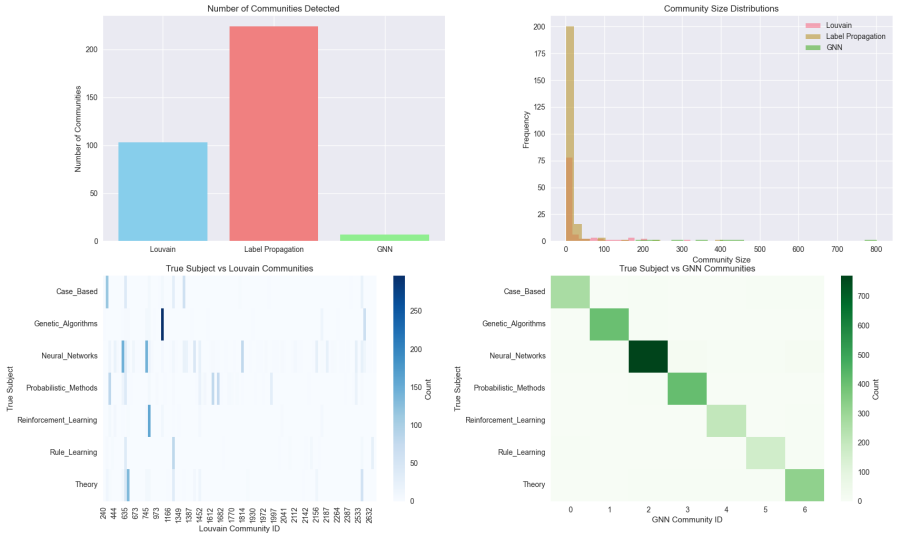


Figure: GCN modeli tarafından öğrenilen düğüm gömülmelerinin (embeddings)

t-SNE ile 2 boyuta indirgenmiş hali

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

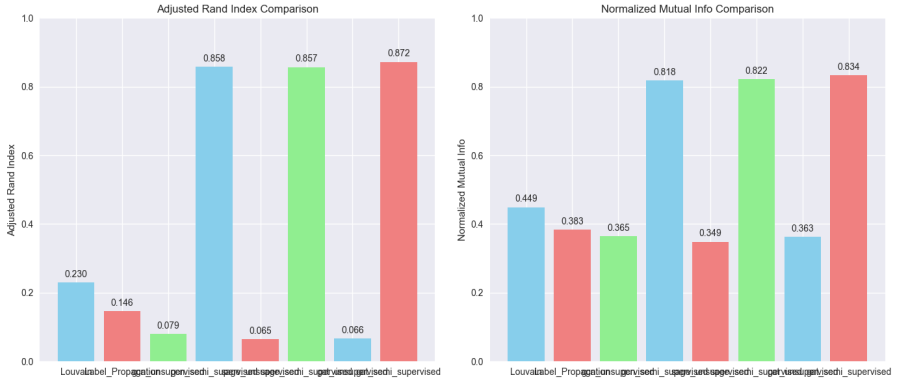


Figure: GraphSAGE modeli tarafından öğrenilen düğüm gömülerinin (embeddings) t-SNE ile 2 boyuta indirgenmiş hali.

Proje Hedefleri ve Araştırma Soruları

Neyi, Neden, Nasıl Sorguladık?

- Klasik topluluk tespit algoritmaları ile modern GNN tabanlı yaklaşımların performansını nicel olarak karşılaştırmak.
- Neo4j GDS üzerinde Louvain ve LPA; PyTorch Geometric ile GCN, GraphSAGE ve GAT modellerini hem denetimsiz hem yarı denetimli senaryolarda incelemek.
- Performansı Modülerlik, ARI, NMI gibi metriklerle değerlendirmek ve gerçek konu etiketleriyle uyumunu ölçmek.
- Büyük ölçekli graph + GNN entegrasyonu için uçtan uca bir araştırma iş akışı sunmak.

Deneyisel Kurulum

Veri, Parametreler ve Donanım

- **Veriseti:** Cora Citation Network (2708 düğüm, 5429 kenar, 1433 özellik, 7 sınıf).
- **GNN Parametreleri:** Gizli boyut = 64, Katman = 2, Öğrenme hızı = 0.01, Epoch = 200.
- **Klasik Algoritmalar:** GDS varsayılan parametreleri (weightless, resolution = 1.0).
- **Donanım:** 16 GB RAM, NVIDIA RTX 3060 GPU (CUDA 11.8).

Temel Bulgular

Özet Değerlendirme Metrikleri

Yöntem	ARI	NMI	Top. Sayısı
Louvain	0.26	0.45	103
Label Propagation	0.15	0.38	224
GCN (Semi)	0.85	0.81	7
GraphSAGE (Semi)	0.85	0.81	7
GAT (Semi)	0.88	0.84	7

- Yarı denetimli GNN modelleri, küçük bir etiketli veri kümesiyle dahi klasik yöntemleri geride bırakmıştır.
- Louvain, yüksek modülerlik üretmesine rağmen gerçek konu etiketleriyle orta düzeyde uyumludur.
- LPA çok sayıda topluluk üreterek gürültüye açık bir yapı göstermiştir.

Çıkarımlar ve Gelecek Çalışmalar

- Yarı denetimli GNN'ler, topluluk tespitinde güçlü bir alternatif sunmaktadır.
- Neo4j + PyG entegrasyonu, araştırmacılar için esnek ve ölçeklenebilir bir altyapı sağlar.
- Gelecekte heterojen graph'lar (R-GCN) ve zamansal topluluk tespiti üzerine odaklanılabilir.

Teşekkürler