ÇANKIRI KARATEKİN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

AĞ YAPILI VERİLER İÇİN ÖZNİTELİK TABANLI ÖZELLİK ÖĞRENİMİ

Sarah Abdulkareem Ahmed AHMED

ELEKTRONİK VE BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

ÇANKIRI
2023

Her hakkı saklıdır

TEZ ONAYI

Sarah Abdulkareem Ahmed AHMED tarafından hazırlanan **"Ağ Yapılı Veriler için Öznitelik Tabanlı Özellik Öğrenimi"** adlı tez çalışması 09/06/2023 tarihinde aşağıdaki jüri tarafından oy birliği ile Çankırı Karatekin Üniversitesi Fen Bilimleri Enstitüsü Elektonik ve Bilgisayar Mühendisliği Anabilim Dalında **Yüksek Lisans Tezi** olarak kabul edilmiştir.

Danışman **:** Doç. Dr. Serkan SAVAŞ

Jüri Üyeleri **:**

Başkan **:** Dr. Öğr. Üyesi Selim BUYRUKOĞLU
Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı
Çankırı Karatekin Üniversitesi

Üye **:** Dr. Öğr. Üyesi Mustafa KARHAN
Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı
Çankırı Karatekin Üniversitesi

Üye **:** Doç. Dr. Serkan SAVAŞ
Bilgisayar Bilimleri Anabilim Dalı
Kırıkkale Üniversitesi

Yukarıdaki sonucu onaylarım

Prof. Dr. Hamit ALYAR

Enstitü Müdürü

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Çankırı Karatekin Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğine göre hazırlamış olduğum "**Ağ Yapılı Veriler için Öznitelik Tabanlı Özellik Öğrenimi**" konulu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı, tezin içerdiği yenilik ve sonuçları başka bir yerden almadığımı, tezde kullandığım eserleri usulüne göre kaynak olarak gösterdiğimi, tezin Çankırı Karatekin Üniversitesi Fen Bilimleri Enstitüsü'nden başka bir bilim kuruluna akademik amaç ve unvan almak amacıyla vermediğimi ve bu çalışmanın Çankırı Karatekin Üniversitesi tarafından kullanılan "Bilimsel İntihal Tespit Programı"yla tarandığını, "intihal içermediğini" beyan ederim. Çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması halinde ortaya çıkacak tüm ahlaki ve hukuki sonuçlara razı olduğumu bildiririm. Çankırı Karatekin Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca gereğinin yapılmasını arz ederim (09/06/2023).

Sarah Abdulkareem Ahmed AHMED

ÖZET

Yüksek Lisans Tezi

AĞ YAPILI VERİLER İÇİN ÖZNİTELİK TABANLI ÖZELLİK ÖĞRENİMİ

Sarah Abdulkareem Ahmed AHMED

Çankırı Karatekin Üniversitesi
Fen Bilimleri Enstitüsü
Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Serkan SAVAŞ

Ağlar, birçok gerçek hayat senaryosunda veriler (düğümler) arasındaki etkileşimleri (bağlantıları) yakalama yeteneklerinden dolayı birçok sistemde yaygın olarak kullanılmaktadır. Ağlar, makine öğreniminde farklı tahmin tabanlı analizler yapmak için kullanılabilir. Ancak mevcut makine öğrenimi algoritmaları, veri örneklerinin birbirinden bağımsız olduğu ve ağ yapılı verilerin düzensiz ve değişken boyutlu doğası nedeniyle ağ bilgilerinin makine öğrenimine dahil edilmesini engellediği yönünde temel bir varsayıma sahiptir. Son zamanlarda, ağdaki yapısal bilgiyi koruyarak her düğüm için düşük boyutlu bir vektör temsilini öğrenmeyi amaçlayan özellik öğrenme yöntemleri önerilmiştir. Bu teknikler yalnızca ağın topoloji bilgisinden faydalanır ve düğüm özniteliklerini dikkate almaz. Ancak gerçek dünya ağları genellikle çok sayıda düğüme ve yüksek boyutlu düğüm özniteliklerine sahiptir. Bu çalışmada, topoloji bilgisinin yanı sıra düğüm özniteliklerinden faydalanmak için rasgele yürüyüş prensibini benimseyen node2vec yönteminin nasıl genişletilebileceği gösterilmiştir. Node2vec'teki bir rasgele yürüyüş, düğüm kimlikleri dizisi olarak temsil edilir, öznitelikleri kullanmak için düğüm kimlikleri dizisi yerine özellik vektörleri dizisi kullanılır. Tek kategorik özniteliğe sahip ağlar için *node2vec-öznitelik*, çok nitelikli ağlar için ise *node2vec-öznitelik+* modelleri önerilmiştir. Önerilen yaklaşımların yapısal rollerin benzerliği ve düğüm sınıflandırması üzerindeki etkinliğini değerlendirmek için gerçek dünya veri setleri üzerinde çeşitli testler gerçekleştirilmiştir. Gerçekleştirilen tüm deneysel çalışmalar, her bir algoritmanın etkinliğini kanıtlamaktadır. Genel olarak sunulan sonuçlar, önerilen modellerin ağ ile ilişkili düğüm içerik bilgilerini yakalama ve kodlama konusundaki etkinliğini ve sağlamlığını ortaya koymaktadır.

2023, 60 sayfa

ANAHTAR KELİMELER: Yapısal veri, Özellik öğrenimi, Ağ temsili öğrenimi, Rasgele yürüyüş, Düğüm sınıflandırma

ABSTRACT

Master of Science Thesis

ATTRIBUTE-BASED FEATURE LEARNING FOR NETWORK-STRUCTURED DATA

Sarah Abdulkareem Ahmed AHMED

Çankırı Karatekin University
Graduate School of Natural and Applied Sciences
Department of Electronics and Computer Engineering

Advisor: Assoc. Prof. Dr. Serkan SAVAŞ

Networks are used widely in various systems because of their capacity to capture the interactions (edges) among data (nodes) in various real-world settings. This ability has led to the widespread adoption of networks. In the field of machine learning, constructed networks may be used to produce different predictive-based analyses. However, current machine learning methods are based on the assumption that data instances are independent of each other and prevent network information from being included in machine learning due to the irregular and variable-size nature of network structured data. Recently, feature learning methods have been proposed that aim to learn a low-dimensional vector representation for each node while preserving the structural information in the network. These techniques only use the network's topology information and do not consider the node attributes. However, real-world networks often have large numbers of nodes and high-dimensional node attributes. This study demonstrates how node2vec, which adopts the random walk principle, can be extended to take advantage of node attributes as well as topology information. With node2vec, a random walk is represented as an array of node identifiers. To utilize attributes, it is expressed as an array of feature vectors. The *node2vec-attribute* model is proposed for networks with a single categorical attribute and the *node2vec-attribute+* model is for multi-attribute networks. Various experiments were performed on real-world datasets to evaluate the effectiveness of the proposed approaches on the similarity of structural roles and node classification. All the experimental studies performed prove the effectiveness of each algorithm. Overall, the presented results demonstrate the effectiveness and robustness of the proposed models in capturing and encoding network-related node content information.

2023, 60 pages

Keywords: Structural data, Feature learning, Network representation learning, Random walk, Node classification

ÖNSÖZ VE TEŞEKKÜR

Tez çalışmam boyunca engin bilgi ve tecrübeleriyle, verdiği destek ve gösterdiği hoşgörüden ötürü değerli danışmanım Doç. Dr. Serkan SAVAŞ‘a, yoğun çalışmalarım sırasında sabır gösterdiği ve bana katlandığı için eşim Sermet ‘e sürekli çalışmalarıma izin verdiği için küçük oğullarım Eren ve Oğuz’a, çalışmalarım sırasında ümit verdiği için kız kardeşim Zeynep’e, çalışmalarım sırasında küçük veya büyük yardımını esirgemeyen herkese teşekkür ederim.

Sarah Abdulkareem Ahmed AHMED

Çankırı, Haziran 2023

İÇİNDEKİLER

ÖZET ..... **i**
ABSTRACT ..... **ii**
ÖNSÖZ VE TEŞEKKÜR ..... **iii**
İÇİNDEKİLER ..... **iv**
SİMGELER DİZİNİ ..... **vi**
KISALTMALAR DİZİNİ ..... **vii**
ŞEKİLLER DİZİNİ ..... **viii**
ÇİZELGELER DİZİNİ ..... **ix**
1. GİRİŞ **1**
2. LİTERATÜR ÖZETİ **6**
3. AĞLAR **10**
3.1 Ağ Analizi **11**
3.2 Düğüm Öznitelik Vektörleri **12**
3.3 Benzerlik **14**
3.4 Ağ Yerleştirme **17**
4. MATERYAL VE METOT **18**
4.1 Ağ Temsili Öğrenimi **19**
4.1.1 Matris çarpanlarına ayırmayı (matrix-factorization) kullanan yöntemler **20**
4.1.2 Olasılık temelli modeller **20**
4.1.3 Derin öğrenme tabanlı yöntemler **22**
4.2 Kelime Yerleştirmeleri **23**
4.3 Node2vec **28**
4.4 Önerilen Model **32**
4.5 Veri Setleri **34**
4.5.1 Topluluk-5 ..... **34**
4.5.2 Email-EU-core **35**
4.5.3 Facebook **36**
4.5.4 Cora **36**
4.6 Temel Yöntemler **36**
4.7 Sınıflandırıcı ve Performans Metriği **37**
5. BULGULAR VE TARTIŞMA **39**
5.1 Yapısal Rollerin Benzerliği **39**

5.2 Düğüm Sınıflandırması 42
5.3 Parametre Hassasiyeti 46
5.4 Tartışma ve Değerlendirme..... 47
6. SONUÇ VE ÖNERİLER 49
KAYNAKLAR 51
ÖZGEÇMİŞ..... 60

SİMGELER DİZİNİ

A	Adjacency matrix (Komşuluk matrisi)
B	Benzerlik ölçüsü
c	window size (Pencere boyutu)
d	Öğrenilen vektör temsilin boyutudur
E	Bağlantı kümesi
$f(u)$	u düğüm noktasının yerleştirme temsili
G	Graph (ağ / çizge)
id	Düğüm tanımlayıcısı (kimliği)
k	Kümeleme sayısı
L	Loss function (Kayıp fonksiyonu)
M	Mesafe ölçüsü
n	Ağdaki toplam düğüm sayısı
$N_S(u)$	u düğümünün ağ komşuluğu
nw	Düğüm başına rasgele yürüyüş sayısı
r_v	v düğümünün öğrenilmiş vektör temsili
S	Örnekleme stratejisi
$\delta(n)$	v_{n} düğüm noktasının özellik vektörü
T	Corpus (Eğitm için oluşan bir dizi cümle kümesi)
t	Önceki düğüm noktası
V	Düğüm kümesi
v_w	Hedef kelimenin (w) vektör temsili
$\tilde{v}_w$	Bağlam (komşu kelimler) vektör temsili
W	Kelime daracığı
wl	Rasgele yürüyüş uzunluğu

KISALTMALAR DİZİNİ

AUC	Area under curve (Eğri altında kalan alan)
BFS	Breadth first search (Genişlik ilk arama)
CBOW	Continous bag of words (Sürekli sözcük torbası)
ctRBM	Conditional temporal restricted boltzmann machine (Koşullu geçici kısıtlı boltzmann makinesi)
DFS	Depth first search (Derinlik ilk arama)
DMF	Discriminative matrix factorization (Ayrımcı matris çarpanlara ayırma)
DNGR	Deep neural networks for graph representations (Ağ temsilleri için derin sinir ağları)
FN	False negative (Yanlış negatif)
FP	False positive (Yanlış pozitif)
GAT	Graph attention networks (Çizge dikkat ağları)
GCN-LPA	Label propagation and graph convolutional neural networks (Etiket yayılımı ve çizge konvolüsyonel sinir ağları)
GENE	Group enhanced network embedding (Grup gelişmiş ağ yerleştirme)
GERI	Graph embedding with rich information (Zengin bilgiler içeren çizge yerleştirme)
GraRep	Graph representation (Çizge temsili)
GW	Guided walk (Rehberli yürüyüş)
HOPE	High-order proximity preserved embedding (Yüksek sıralı yakınlık korumalı yerleştirme)
IsoMap	Isometric feature mapping (İzometrik özellik eşleme)
LINE	Large-scale information network embedding (Büyük ölçekli bilgi ağı yerleştirme)
LLE	Locally linear embedding (Yerel doğrusal yerleştirme)
LR	Logistic regression (Lojistik regresyon)
MDS	Multidimensional scaling (Çok boyutlu ölçekleme)
NLP	Natural language processing (Doğal dil işleme)
SDAE	Stacked denoising autoencoder (Gürültü giderici otomatik kodlayıcıyı)
SDNE	Structural deep network embedding (Yapısal derin ağ yerleştirme)
SVM	Support vector machine (Destek vektör makinesi)
TF - IDF	Term frequency - inverse document frequency (Terim sıklığı - ters belge sıklığı)
TLINE	Transductive large-scale information network embedding (Transdüktif büyük ölçekli bilgi ağı yerleştirme)
TP	True positive (Doğru pozitif)

ŞEKİLLER DİZİNİ

Şekil 1.1 $\boldsymbol{v1}$ ve $\boldsymbol{v2}$ düğümleri için örnek bir atıf ağı 4
Şekil 3.1 5 düğüm ve 9 kenardan oluşan bir ağ 10
Şekil 3.2 Farklı benzerlik türlerini gösteren örnek bir ağ 16
Şekil 3.3 Zachary Karate Club ağı için düğüm yerleştirmeleri (Perozzi et al. 2014) 17
Şekil 4.1 Önerilen modelin düğüm yerleştirme süreci 18
Şekil 4.2 window_size'ın 2 olduğu bir örnek 24
Şekil 4.3 Skip-gram modeli için oluşturulan veri örneği 26
Şekil 4.4 Skip-gram modelinin mimarisi 27
Şekil 4.5 Node2vec algoritmasında geçiş olasılığının nasıl hesaplanması (Grover and Leskovec 2016) 31
Şekil 4.6 Node2vec modelin ardışık düzeni 31
Şekil 4.7 Öznitelik bilgisi içeren örnek bir ağ 32
Şekil 4.8 Önerilen yöntemin (node2vec-öznitelik) ardışık düzeni 33
Şekil 4.9 Çoklu özniteliğe sahip ağlar için önerilen yöntemin (node2vec-öznitelik+) ardışık düzeni 34
Şekil 4.10 Beş topluluğa sahip Topluluk-5 ağı 35
Şekil 5.1 Node2vec ile k-ortalama (k=4) kümeleme sonucu 40
Şekil 5.2 Node2vec ile elde edilen düğüm temsillerin iki boyutlu gösterimi 41
Şekil 5.3 Node2vec-öznitelik ile k-means (k=4) kümeleme sonucu 41
Şekil 5.4 Node2vec-öznitelik ile elde edilen düğüm temsillerin iki boyutlu gösterimi 42
Şekil 5.5 Eksik etiketler için düğüm sınıflandırması 43
Şekil 5.6 Node2vec tabanlı yöntemlerin yürüyüş uzunluğu ve yürüyüş sayısı parametlerinin değişimindeki performansı 46

ÇİZELGELER DİZİNİ

Çizelge 4.1 Topluluk-5 ağında düğüm renklerinin atanması 35
Çizelge 4.2 Veri setlerinin istatistiklerine genel bir bakış 36
Çizelge 5.1 Topluluk-5 ağındaki düğümlerin yapısal rolleri 39
Çizelge 5.2 Node2vec tabanlı yöntemler için seçilen $\boldsymbol{p}$ ve $\boldsymbol{q}$ değerleri 43
Çizelge 5.3 Tek özniteliğe sahip ağların düğüm sınıflandırma performansı 44
Çizelge 5.4 Cora ağı için farklı yöntemlerin düğüm sınıflandırma performansı 45
Çizelge 5.5 Cora ağı için düğüm sınıflandırma görevinde model başarı sonuçları 47

1. GİRİŞ

Ağlar, günümüzde karmaşık sistemlerin içerdiği unsurların birbirleriyle olan ilişkilerini graf yapıları ile temsil etmede kullanılmaktadır (Barrat *et al.* 2008). Bir ağı oluşturan unsurlar köşeler (düğümler) ve kenarlardır (bağlantılardır). Köşeler varlıkları temsil ederken kenarlar varlıklar arasındaki ilişkilerdir. Elektrik şebekelerini ifade eden ağlar, sinir ağları, bilimsel iş birliği ağları, spor ağları, ulaşım ağları, internet ağı, arkadaşlık ağları gibi pek çok gerçek dünya uygulamaları ağların kullanıldığı alanlara örnek gösterilebilir. Gerçek dünya uygulamalarının ağlarla gösterimi yaygın olarak kullanılmaktadır. İletişim teknolojisindeki gelişmeler ile özellikle sosyal ağların yaygınlaşması ağların kullanımının önemini arttırmıştır. Bu ağların ölçeği binlerce, yüzlerce ve hatta milyonlarca düğüm arasında değişmektedir (Tang *et al.* 2015).

Dünya çapında ağa (World Wide Web) hâkim olan başlıca ağ modeller, hayatımızın ayrılmaz bir parçası haline gelen ve insanların birbirleriyle ve dünyayla etkileşim biçimlerini önemli ölçüde değiştiren sosyal ağlar olmuştur. Günümüzün sosyal ağları, kullanıcıları hakkında ilgili düşünceleri elde etmek için incelenebilecek çok sayıda değerli veriyle doludur. Bu veriler daha sonra bir kullanıcıya yeni arkadaşlar önermek veya bir kullanıcı profilindeki eksik bilgileri tahmin etmek (Nandi and Das 2014) gibi çeşitli uygulamalarda kullanılabilir. Biyoinformatikte, hastalıklar için yeni tedavileri kolaylaştırmak amacıyla analiz edilebilen protein-protein etkileşimlerini modellemek için ağlar kullanılabilir (Moschopoulos *et al.* 2009). Literatürde yayınlar ise, bir yayın diğer yayına atıfta bulunuyorsa, iki yayının bir kenarla birleştirildiği bir ağ şeklinde temsil edilebilir. Bu tür ağlar, çeşitli yayınlar arasındaki benzerlik, ilişkileri bulmak ve bunları farklı kategorilere ayırmak için analiz edilebilir (Brin 1998). Gerçek dünya ağlarını analiz etmek, birçok disiplinde ortaya çıkan çeşitli uygulamalarda önemli rol oynamaktadır. Örneğin, sosyal ağlarda, kullanıcıları anlamlı sosyal gruplara ayırmak, kullanıcı arama ve hedefli reklamcılık gibi birçok önemli görev için yararlıdır; iletişim ağlarında, topluluk yapılarını tespit etmek dedikodu (söylenti) yayma sürecini daha iyi anlamaya yardımcı olabilir.

Herhangi iki düğüm arasında gelecekteki bir bağlantıyı tahmin etmek, yeni bağlantılar önermek, bilinmeyen düğümleri etiketlemek ve düğüm verilerini kümelemek için uygun araçlar kullanılarak ağlar analiz edilebilir. Yukarıda belirtilen uygulamaların etkinliği, ağ analizinde kullanılan yöntemlerin kalitesine ve doğruluğuna bağlıdır (Chami *et al.* 2022).

Gerçek dünya ağların büyük boyutu ve dinamik doğası nedeniyle, ağ analizi için geleneksel yöntemlerin etkisiz olduğu gösterilmektedir. Ağ analizinin performansını artırmak için makine öğrenimi tabanlı veri analizi tekniklerin kullanma ihtiyacı artmaktadır. Bununla birlikte, makine öğrenimi teknikleri, ızgaralar (grids) (örneğin görüntüler) ve diziler (örneğin metin, konuşma) gibi Öklid uzaylarında tanımlanan verilerle çalışacak şekilde tasarlanmıştır ve ağlar gibi Öklid uzaylarında tanımlanamayan verilerle doğrudan kullanılamaz (Bronstein *et al.* 2017). Bu sınırlamanın üstesinden gelmek için, ağ bilgilerini mevcut makine öğrenimi modelleri tarafından işlenebilecek bir temsile dönüştürecek bir yöntem gereklidir.

Ağlardaki bilgileri eşdeğer bir Öklid uzayına dönüştürülürken, ağdaki yapısal bilgilerin olabildiğince korunarak dönüştürme kaybının en aza indirilmesi önemlidir (Hamilton *et al*. 2017). Dönüştürülen temsile yerleştirme denir ve sürece düğüm yerleştirme denir. Düğüm yerleştirmeleri oluşturulduktan sonra, düğüm sınıflandırma, bağlantı tahmini ve topluluk tespiti gibi görevler için makine öğrenimi teknikleriyle birlikte kullanılabilirler. Bu görevlerin başarısı ve ağların etkili analizi, büyük ölçüde oluşturulan yerleştirmelerin kalitesine bağlıdır (Mohammed and Gündüç 2022).

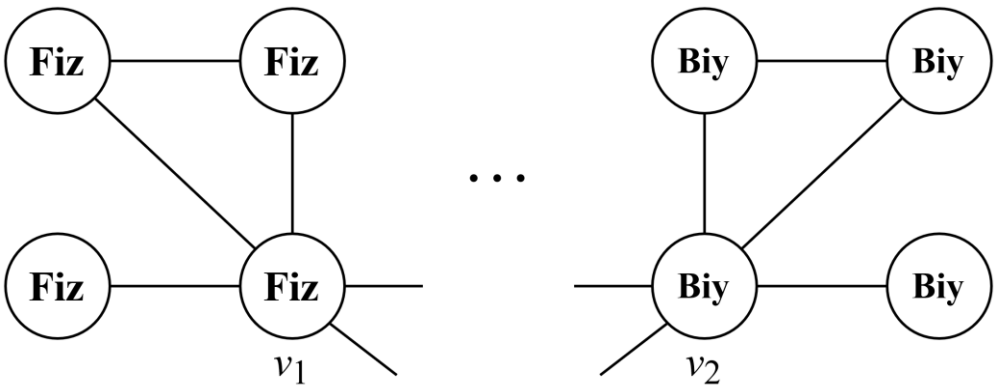
Genellikle bir ağın temsili, yalnızca düğümler arasındaki komşu ilişkilerini yakalayan ayrık bir komşuluk matrisi ile gösterilir. Bu basit komşuluk matris temsili, yollar, altçizge vb. gibi daha karmaşık, daha üst düzey yapı ilişkilerini içeremez. Bu nedenle, birçok ağ analiz görevi, bu tür geleneksel bir rutine bağlı kaldığında, büyük ölçekli ağlarda hesaplama açısından maliyetli ve zor hale gelecektir. Örneğin, topluluk tespiti için mevcut yöntemlerin çoğu, en az ikinci dereceden zaman karmaşıklığı ile bir matrisin spektral ayrışmasının hesaplanmasını gerektirir (Kumar and Hanot 2021). Bu hesaplama yükü, yöntemleri büyük ağlara ölçeklendirmeyi zorlaştırır.

Son zamanlarda yapılan araştırmalar, ağ temsili öğrenimi üzerine odaklanmıştır. Ağ temsili öğrenimi, ağ topoloji yapısını ve/veya düğüm içeriğini korurken ağ düğümlerinin düşük boyutlu vektör temsillerini öğrenmeyi amaçlar. Bu nedenle, orijinal ağ verilerine uygulanan karmaşık algoritmaların türetme ihtiyacını ortadan kaldırır.

Önceki ağ yerleştirme çalışmaları, araştırmaların boyutluluk azaltma stratejilerinin bir bileşeni olarak ağ yerleştirme algoritmalarını ilk kez önerdiği 2000'li yılların başına kadar uzanır. Girdi olarak bir dizi veri noktası verildiğinde, ağ yerleştirme teknikleri önce, örneğin k-en yakın komşu ağı gibi bir yakınlık ağı oluşturmak için çift veri noktaları arasındaki benzerliği hesaplar ve ardından bunu daha düşük boyutlu bir uzaya dönüştürür. Amaç, ağın yüksek boyutlu veri geometrisinde gizlenmiş düşük boyutlu bir manifold yapısını ortaya çıkarmaktır, böylece bağlı düğümler yeni yerleştirme uzayında birbirine daha yakın tutulur (Chami *et al.* 2022). İzometrik özellik eşleme (isometric feature mapping – IsoMap) (Tenenbaum *et al.* 2000), yerel doğrusal yerleştirme (locally linear embedding – LLE) (Roweis and Saul 2000) ve Laplacian Eigenmap (Belkin and Niyogi 2001) bu mantığa dayanan algoritma örnekleridir. Bu algoritmaların çoğu, düğüm sayısına göre ikinci dereceden zaman karmaşıklığına sahiptir. Dolayısıyla, büyük ölçekli ağlarda kullanıldığında, ölçeklenebilirlik bu çözümler için önemli bir zorluk haline gelir.

2010'den bu yana, gerçek dünya ağları için tasarlanmış verimli ve ölçeklenebilir temsili öğrenme yöntemleri oluşturmaya odaklanan birçok araştırma yapılmıştır. (Perozzi *et al.* 2014, Tang *et al.* 2015, Cao *et al.* 2016, Grover and Leskovec 2016, Zhang *et al.* 2016a) gibi birçok ağ yerleştirme algoritması, gerçek dünya ağları yerleştirmek için önerilmiştir ve çeşitli uygulamalar için umut verici performans göstermektedir. Bu yaklaşımlar, yapısal yakınlığı koruyan düşük boyutlu bir alana ağı yerleştirir, böylece ağdaki orijinal düğümler düşük boyutlu vektörler olarak temsil edilir. Ortaya çıkan düşük boyutlu vektör temsilleri, makine öğrenme yöntemlerinde özellik olarak kullanılabilir. Sonuç olarak, düğüm sınıflandırması (Bhagat *et al.* 2011), topluluk tespiti (Malliaros and Vazirgiannis 2013), bağlantı tahmini (Lü and Zhou 2011) ve görselleştirme (Tang *et al.* 2016) dahil olmak üzere çeşitli ağ analiz görevlerini artık yeni vektör uzayında kolayca ve verimli bir şekilde gerçekleştirilebilir.

Mevcut ağ yerleştirme yöntemleri, düğüm temsillerini öğrenmek için yalnızca ağ yapısından faydalanır, böylece önemli yapısal ilişkililiğe sahip düğümler yeni yerleştirme alanında birbirine yakın temsil edilir. Çoğu durumda, ağ yapısı, düğüm benzerliğini ölçmek için gürültülü, seyrek ve bilgi açısından yetersizdir. Sonuç olarak, yalnızca yapıyı koruyan ağ yerleştirme algoritmaları, ağ analiz uygulamaları için yüksek kaliteli düğüm temsillerini öğrenmeyebilir. Ağ yapısının yanı sıra, daha iyi ağ yerleştirmesi öğrenebilmek için düğüm içerik düzeyindeki benzerliği ölçmek için düğüm öznitelikleri kullanılabilir. Örneğin, sosyal ağlarında yaş ve ilgi alanları veya atıf ağlarında anahtar kelimeler ve yayın tarihi gibi birçok farklı öznitelik gerçek dünya ağlarında bulunabilir. Belirli problemler için, bu tür önemli verilerin ağın yapısı ile birlikte kullanılması çok önemlidir. Örneğin, Şekil 1.1'deki atıf ağının v_1 ve v_2 düğümleri ele alınırsa. Yalnızca ağ yapısından yararlanan bir ölçü kullanılıyorsa, v_1 ve v_2 düğümleri azami derecede benzer olduğunu bildirir. Ancak, düğüm özniteliklere bakılırsa, bunların farklı olduğu görülebilir. Belki de bu öznitelikler, biri fizikte, bir diğeri biyolojide olmak üzere bu makalelerin yayınlandığı alanı tanımlar. Göreve bağlı olarak, biyoloji yayınları olan v_2'ye benzer yayınlar bulmak istenebilir. Dolayısıyla, bu durumda sadece yapıya bakmak, yapı ve özniteliklerin bir kombinasyonunun sahip olabileceği tatmin edici bir sonuç vermez.



Şekil 1.1 v_1 ve **v_2** düğümleri için örnek bir atıf ağı

Düğüm temsillerini öğrenmek için ağ yapısını koruyan mevcut yaklaşımların çoğu, düğüm içerik özelliklerinde bulunan önemli bilgileri göz ardı ederek yalnızca ağ yapısını kullanır. Dolayısıyla, öğrenilen düğümlerin daha az bilgilendirici temsilleriyle sonuçlanmaktadır.

Bu tez çalışmasında, ağ yapılı verilerde düğümlerin düşük boyutlu özellik temsilleri öğrenmesi amaçlanmıştır. Bu düşük boyutlu temsile yerleştirme vektörü denir. Tezde önerilen çalışma, ağlar üzerinde hem rasgele yürüyüş tabanlı hem de yapay sinir ağları tabanlı yöntemleri kullanan önceki düğüm yerleştirme tekniklerini kavramsal olarak temel almaktadır. Bu prensibi benimseyen node2vec yöntemi bu çalışmada ayrıntılı olarak ele alınmıştır. Çalışmanın motivasyonu, ağ yapısı ve düğüm öznitelikleri arasındaki işbirliği yoluyla, ağlar için bilgilendirici düğüm temsillerini öğrenmeyi amaçlamaktadır. Ağ temsillerini öğrenmek için ağın yapısal ve düğüm içeriğinden gelen bilgileri etkili bir şekilde kullanmasına izin veren iki farklı node2vec uzantısı önerilmiştir. Önerilen yaklaşımlar, bir ağı etkili bir şekilde iki bilgi kaynağı arasındaki etkileşimi yakalayan tek bir düşük boyutlu temsil alanına yerleştirmektedir. Öğrenilen temsiller hem ağ yapısını hem de düğüm içerik bilgilerini korumaktadır. Böylece daha fazla bilgi içe aktarıldıkça, çalışmada önerilen ağ yerleştirme yöntemlerin, yalnızca ağ yapısını koruyan tek yapılı yöntemlerden daha iyi performans gösterebildiği kanıtlanmıştır. Önerilen yaklaşımların yapısal rollerin benzerliği ve düğüm sınıflandırması üzerindeki etkinliğini değerlendirmek için 4 farklı ağ veri setleri üzerinde çeşitli testler gerçekleştirilmiştir. Genel olarak sunulan sonuçlar, önerilen modellerin ağ ile ilişkili düğüm içerik bilgilerini yakalama ve kodlama konusundaki etkinliğini ve sağlamlığını ortaya koymaktadır.

Tezin geri kalan bölümleri şu şekilde özetlenmiştir:

- Bölüm 2, klasik ve modern ağ temsili öğrenme ile ilgili çalışmaların literatür incelemesini sunmaktadır.
- Bölüm 3, önemli terminolojilerin tanımlarını ve ağ temsili öğrenme probleminin yanı sıra tezde kullanılan benzerlik ölçütlerinin özetlenmesini sağlar.
- Bölüm 4, veri kümelerini, uygun değerlendirme ölçütlerini açıklar. Ayrıca, önerilen modelin düğüm özniteliklerin yapısal bilgiyle birlikte ortak bir amaç fonksiyonu aracılığıyla nasıl kodlanabileceğini sunmaktadır.
- Bölüm 5, önerilen modeli farklı ağ analiz görevleri için 4 ağ veri seti üzerinde temel yöntemlerle karşılaştırır ve sonuçları hakkında bilgi verir.
- Son olarak, bölüm 6'da çalışmamızın özetini vererek tezimiz sonuçlandırılmıştır ve gelecekteki araştırma yönlerine işaret edilmişitr.

2. LİTERATÜR ÖZETİ

Menon ve Elkan (2011), ağ yapısı için bir matris çarpanlara ayırma modeli önererek, bu tekniği bağlantı tahmin yöntemine başarıyla uygulayan ilk araştırmacılar olmuşlardır. Model, kenarların gizli özelliklerini ağın topolojik yapısından öğrenerek açık özelliklerle birleştirmektedir. Ayrıca, bağlantı tahmini veri setlerinde sık görülen dengesizlik sorununun üstesinden gelmek için model bir sıralama kaybıyla (ranking loss) eğitilmiştir. Adamic-Adar ve Katz gibi popüler gözetimsiz sınıflandırıcılara göre önerilen modelin performansı yaklaşık %10 artırabileceğini göstermişlerdir (Menon and Elkan 2011).

Tang *et al.* (2015) yaptıkları çalışmada, yönlü, yönsüz ve/veya ağırlıklı bilgi ağları için uygun olan bir ağ yerleştirme yöntemi önermişlerdir. Önerilen model, sırasıyla 1 adımlı ve 2 adımlı düğüm ilişkilerini yakalamak için iki farklı yakınlık modeli sunmuştur. Bir kenarı paylaşan iki düğüm, yeni yerleştirme uzayında yakından temsil edilme eğiliminde oldukları birinci dereceden yakınlık olarak kabul edilir. Birçok komşu düğümü paylaşıyorlarsa, iki düğümün benzer gizli temsillere sahip olma eğiliminde olduğu ikinci dereceden yakınlık olarak kabul edilir. İki model tarafından öğrenilen düğüm temsilleri son düğüm temsilleri olarak birleştirilir. Dil ağları, sosyal ağlar ve alıntı ağları dahil olmak üzere çeşitli gerçek dünya ağları üzerindeki deneysel sonuçlar, önerilen modelin verimliliğini ve etkinliğini kanıtlamıştır (Tang *et al.* 2015).

Grover ve Leskovec (2016) çalışmalarında, orijinal ağdan hem lokal hem de global topoloji bilgilerini dikkate alarak düğüm dizilerini örneklemek için taraflı rasgele yürüyüşler kullanmışlardır. Bir düğümün çevresine genişlik ilk arama (breadth first search - BFS) veya derinlik ilk arama (depth first search - DFS) keşfini simüle etmek için yürüyüşü kontrol eden iki parametre (p ve q) kullanarak Skip-gram'ı taraflı rasgele yürüyüşlerle birleştirmişlerdir. Model, d boyutlu özellik uzayında düğümlerin ağ komşuluklarını koruma olasılığını maksimize eden özellik temsillerini öğrenmiştir. Çalışmada, sosyal ağlar, bilgi ağları ve biyoloji ağlar gibi çeşitli gerçek dünya ağları ile deneyler yapılmıştır. Deneyler, önerilen modelin temel yöntemlerden çok etiketli sınıflandırmada %26,7'ye kadar ve bağlantı tahmininde %12,6'ya kadar daha iyi performans göstermiştir (Grover and Leskovec 2016).

Tu *et al.* (2016) çalışmalarında sosyal ağlar için ayrımcı bir temsili öğrenme modeli sunmuşlardır. Çalışma, DeepWalk'ın matris çarpanlarına ayırma versiyonunun (Yang *et al.* 2015) hedefini çok sınıflı destek vektör makinesi (support vector machine - SVM) hedefi ile birleştirmiştir. Bu iki hedef terimini dönüşümlü olarak optimize ederek, eğitim sırasında hem ağ yapısından hem de düğümlerin etiketlerinden gelen bilgileri eş zamanlı olarak birleştirmiştir. Çalışmada öğrenilen temsillerin görselleştirmeleri denetimsiz modellere göre daha ayrımcı olduğunu göstermiştir ve düğüm sınıflandırmasına ilişkin deneysel sonuçlar, önerilen yöntemin diğer modern yöntemlere göre doğruluk açısından önemli bir gelişme sağladığını göstermiştir (Tu *et al*. 2016).

Perrozi *et al.* (2017) yaptıkları çalışmada, ağdaki düğümlerin çok ölçekli temsillerini öğrenmek için yeni bir yaklaşım önermişlerdir. Önceki çalışmaların aksine, bu temsiller, çok ölçekli düğüm ilişkilerini analitik olarak türetilebilir bir şekilde kodlanmıştır. Önerilen yaklaşım, devam eden yürüyüşte birden fazla düğümü atlayarak DeepWalk'ın rasgele yürüyüş stratejisini değiştirmiştir. BlogCatalog, DBLP, Flickr ve YouTube gibi sosyal ağlar için çok etiketli ağ sınıflandırma görevlerinde modelin etkinliğini değerlendirilmiştir. Sonuçlar, çok etiketli sınıflandırma görevlerinde önerilen modelin DeepWalk'tan %10'a kadar ve büyük ölçekli bilgi ağı yerleştirmeden (large-scale information network embedding – LINE) %58'e kadar daha iyi performans göstermiştir (Perozzi *et al.* 2017).

Singer *et al*. (2019) çalışmalarında zamansal ağ düğümlerinin ve kenarlarının zaman içindeki gelişimini öğrenen ve bu dinamikleri farklı ağ tahmin görevleri için geçici bir düğüm yerleştirme çerçevesine dahil eden bir yaklaşım önermişlerdir. Bu çalışma aynı zamanda bir ortak kayıp fonksiyonu sunmuştur, bu fonksiyon verilen görev başına optimizasyonu kullanarak bir düğümün geçici olarak yerleştirmesini oluşturmuştur. Yaklaşım, statik düğüm yerleştirmeleriyle başlar ve bunları ortak bir optimizasyonda farklı zaman noktalarında kullanır. Yaklaşımın etkinliği, düğüm sınıflandırması ve bağlantı tahmini olmak üzere, farklı zaman aralıkları ve ayrıntı düzeyine sahip, ağırlıklı ve ağırlıksız, yönlü ve yönsüz, özellikle çeşitli büyük ölçekli zamansal ağlar üzerinde değerlendirilmiştir (Singer *et al*. 2019).

Wang *et al*. (2020) çalışmalarında ağ analizi için GraphSAGE, node2vec ve metapath2vec ağ yerleştirme yöntemlerini kullanmışlardır. Çalışmaları, modellerin denetimsiz bir şekilde eğitildiğini göstermiştir. Yöntemlerden ikisi, patent ağlarındaki yapısal bilgileri kullanır: homojen ağlar için node2vec ve heterojen ağlar için metapath2vec. Bu çalışmanın sonucu, özellikle düşük kümeleme katsayısına sahip ağlarda node2vec'in GraphSAGE ve metapath2vec'den daha iyi performans göstermiştir (Wang *et al*. 2020).

Salamat *et al*. (2020) yaptıkları çalışmada, ağ yapısına uyum sağlayan dengeli rasgele yürüyüşler kullanarak ağlardaki düğümler için özellik temsillerini öğrenmeyi amaçlamıştır. Çalışmanın amacı, rasgele bir yürüyüş kullanılırken her bir düğümün yürüme olasılıklarını hesaplamada örnekleme sorunlarını önlemek ve bunları gerçek derece dağılımı ile karşılaştırmak için yeni bir yaklaşım geliştirmiştir. Deneysel sonuçlar, önerilen modelin 5 farklı veri setinde diğer yöntemlere göre en yüksek eğri altında kalan alan (area under curve - AUC) skorları kazandığını göstermiştir (Salamat *et al.* 2020).

Fazaeli ve Momtazi (2022) çalışmalarında düğüm özniteliklerine sahip olmayan ağlarda yarı gözetimli Guided-Walk modelini önermişlerdir. Önerilen model, aynı sınıfa ait düğümleri birbirine bağlayan yollar üzerinde daha fazla vurgu yaparak ağ yollarını keşfetmeye dayanmaktadır. Dolayısıyla, seçilen yollar doğru gizli özelliklerin yayılma şansını artırarak düğüm sınıflandırma görevi için uygun bir seçenek olmuştur. Cora, Pubmed, Twitch DE ve Facebook veri kümeleri üzerinde yapılan deneyler, en yeni tekniklere göre düğüm sınıflandırma doğruluğunda sırasıyla %0,53, %0,78, %5,07 ve %7,13'lük iyileşme göstermiştir (Fazaeli and Momtazi 2022).

Yu *et al*. (2022) çalışmalarında karmaşık ağların bağlantı tahmininde derece, kümeleme katsayısı ve düğüm merkeziyetini birleştiren bir bağlantı tahmin algoritması önermişlerdir. Önerilen model, ağdaki düğümlerin önemini göz önünde bulundurmak için düğüm derecesini, kümeleşme katsayısı ve yakınlık merkeziyetini kullanarak yerel bilgiyi dikkate almaktadır. Yerel benzerliğe dayanan yöntemin avantajı düşük hesaplama karmaşıklığıdır. Bununla birlikte, sınırlı kullanılan bilgi nedeniyle tahmin doğruluğu ideal değildir (Yu *et al.* 2022).

Node2vec yönteminin stokastik gradyan iniş yöntemi, önceden ağ bilgisi eksikliği nedeniyle orta düzeyde bir yerel optimum değerine düşme eğilimindedir ve bu durum ağın genel yapısını yakalayamama sorununa yol açar. Bu sorunu aşmak için, Khanam *et al*. (2023) yaptıkları çalışmada, node2vec ve derin öğrenme yönteminin entegrasyonuna imkan sağlayan bir derin öğrenme modeli önermişlerdir. Önerilen model, node2vec tarafından çıkarılan özellikleri dört katmanlı gizli sinir ağına besleyen bir yapıya sahiptir. Ayrıca, önerilen model bağlantı tahmininin performansını artırmak için Adam, Adamax, Adadelta ve Adagrad gibi adaptif öğrenme optimize edicilerinden yararlanmaktadır. Yapılan deneysel sonuçlar, önerilen yöntemin çeşitli sosyal ağ veri kümelerinde mevcut yöntemlere göre daha iyi sonuçlar verdiğini göstermiştir (Khanam *et al.* 2021).

Ağ verilerinde büyük miktarda gürültü bulunması, özellik çıkarımını etkiler. Çizge dikkat ağları (graph attention networks – GAT) modeli gürültülü verilere duyarlı olduğu için Xie *et al*. (2023) yaptıkları çalışmada, GAT modelini yeni kodlayıcı olarak entegre etmek için adaptif ağırlıklar kullanmayı mümkün kılan yeni bir model önermişlerdir. Önerilen modelin etkinliği, üç genel veri seti (Cora, Citeseer ve Pubmed) üzerinde yapılan optimizasyonlarla link tahmin görevinde %2,2'ye kadar ve düğüm kümeleme görevinde %12,9'a kadar iyileştirme göstermiştir (Xie *et al.* 2023).

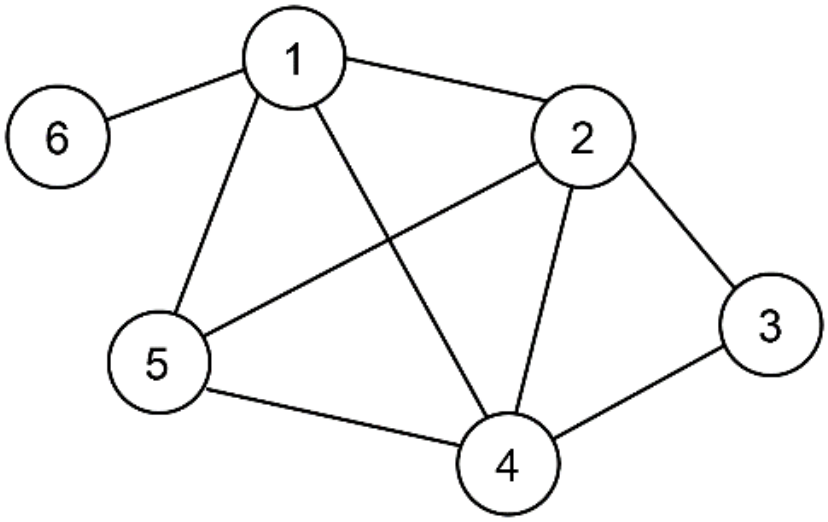
Mevcut ağ yerleştirme yöntemleri, düğüm temsillerini öğrenmek için yalnızca ağ yapısından faydalanır, böylece önemli yapısal ilişkililiğe sahip düğümler yeni yerleştirme alanında birbirine yakın temsil edilir. Çoğu durumda, ağ yapısı, düğüm benzerliğini ölçmek için gürültülü, seyrek ve bilgi açısından yetersizdir. Sonuç olarak, yalnızca yapıyı koruyan ağ yerleştirme algoritmaları, ağ analiz uygulamaları için yüksek kaliteli düğüm temsillerini öğrenmeyebilir. Ağ yapısının yanı sıra, daha iyi ağ yerleştirmesi öğrenebilmek için düğüm içerik düzeyindeki benzerliği ölçmek için düğüm öznitelikleri kullanılabilir. Bu çalışma, ağ yapısı ve düğüm öznitelikleri arasındaki işbirliği yoluyla, ağlar için bilgilendirici düğüm temsillerini öğrenmeyi amaçlamaktadır. Böylece daha fazla bilgi içe aktarıldıkça, çalışmada önerilen ağ yerleştirme yöntemlerin, yalnızca ağ yapısını koruyan tek yapılı yöntemlerden daha iyi performans gösterebildiği kanıtlanmıştır.

3. AĞLAR

Bilgisayar dünyasında bulunan ve gerçek hayatta çeşitli sebeplerle karşılaşılan yapıları temsil amacıyla kullanılan şekillerdir. Örneğin bir karar ağacını ağ kullanarak temsil etmek mümkündür. Bilgisayar bilimleri çeşitli uygulamalarda karşılaşılan bu yapıları ifade etmek için çeşitli matematiksel ve görsel yöntemlerden faydalanır. Buna göre bir ağda bulunan varlıklar düğümler ile ifade edilmekte, bu varlıklar arasındaki ilişkiler ise ağdaki kenarlar ile ifade edilmektedir. Kenarların yönlü olup olmamasına göre, yönlü ağlar ve yönsüz ağlar olarak ikiye ayırmak mümkündür. Ayrıca kenarların değer almasına göre değerli ağlar veya değersiz ağlar isimleri verilebilir (Seker 2008).

Bir ağ, kenarlarla birbirine bağlı bir düğümler koleksiyonudur. Bir ağ, $G = (V, E)$ olarak temsil edilebilir; burada V düğümler (köşeler) kümesidir ve E her bir kenarın iki düğümü birleştirdiği bir kenarlar (bağlantılar) kümesidir. 6 düğümü ve 9 kenarı olan bir ağ örneği Şekil 3.1'de gösterilmiştir.

$V = \{1, 2, 3, 4, 5, 6\}$
$E = \{(1, 2), (1, 4), (1, 5), (1, 6), (2, 3), (2, 4), (2, 5), (3, 4), (4, 5)\}$



Şekil 3.1 5 düğüm ve 9 kenardan oluşan bir ağ

Ağlar, çeşitli şekillerde temsil edilebilen veri yapılarıdır. Bir ağ temsili, tüm kenarların bir listesi veya bir kenarın varlığının 1 ve yokluğunun 0 olduğu bir $(n \times n)$ komşuluk matrisi şeklinde ifade edilir. Burada n, ağdaki düğüm sayısıdır. Şekil 3.1'deki ağ için komşuluk matrisi şu şekilde verilir:

$\boldsymbol{A}$ =

	1	**2**	**3**	**4**	**5**	**6**
1	0	1	0	1	1	1
2	1	0	1	1	1	0
3	0	1	0	1	0	0
4	1	1	1	0	1	0
5	1	1	0	1	0	0
6	1	0	0	0	0	0

$$A_{ij} = \begin{cases} 1 & eğer\ i\ ve\ j\ düğümleri\ arasında\ bir\ bağlantı\ varsa, \\ 0 & aksi\ takdirde \end{cases}$$

3.1 Ağ Analizi

Ağ bilimi, ağı oluşturan varlıkları, varlıklar arasındaki ilişkileri ve ağdaki veri akışını belli kurallar ve disiplinler çerçevesinde analiz edebilmeye imkan sağlamaktadır. Günümüzde ağ analizi çalışmaları yaygın olarak sosyal ağların analizi üzerine yoğunlaşmış olsa da hemen hemen her alandan ilgi görmektedir (Savaş ve Topaloğlu 2018, Knoke and Yang 2019, Savaş 2022). Ağ analizi teknikleri ilk başlarda sosyal ağ analizi üzerinde kullanılırken günümüzde spor, ulaşım, eğitim, bankacılık, biyoloji, gibi daha karmaşık yapıya sahip alanlarda da uygulanmaktadır (Boffey 1982, Mason and Verwoerd 2007, Likaj *et al.* 2013).

Gerçek dünya ağlarının incelenmesi ile ilgili tarihsel sürece bakıldığında, İsviçreli matematikçi Leonhard Euler'in 1736'da çizge teorisiyle çözdüğü Konigsberg Köprüsü problemi bu alanda bilinen ilk çalışma olarak dikkat çekmektedir (Euler 1956). 1950'den sonra Erdős–Rényi tarafından önerilen rassal ağlar bu alanda çalışmaların

yaygınlaşmasında önemli rol oynamıştır (Erdős and Rényi 1960). Özellikle bilgisayar ve iletişim teknolojisinin hızla gelişmesiyle daha büyük ve karmaşık gerçek dünya ağlarının analizi ilgi gören araştırma konuları arasında yerini almıştır. Ağlarla ilgili çalışmalar gerçek dünya ağlarının ortak özelliklerinin ortaya çıkarılmasına katkı sağlamıştır. Elde edilen bu özellikler sonraki dönemlerde geliştirilerek, karmaşık yapıları anlamamız için yapılan çalışmalara katkı sunmuştur (Zhou *et al.* 2022).

Ağ analizi çalışmalarına bakıldığında, temel olarak ağların istatistiksel analizi (Huisman and Van Duijn 2004), topluluk algılama ve düğümlerin sınıflandırılması (Kumar and Hanot 2021), ağ dinamiklerinin zaman içerisindeki değişimi (Kumar *et al.* 2006), bilgi yayılımı (Gomez-Rodriguez *et al.* 2012, Stieglitz and Dang-Xuan 2013), sosyal medya siber istihbaratı (Savaş ve Topaloğlu 2015, 2016), ağlar üzerinde veri madenciliği ve ağların görselleştirilmesi (Breitkreutz *et al.* 2003, Xia *et al.* 2013) üzerine yoğunluk kazandığı görülmektedir.

3.2 Düğüm Öznitelik Vektörleri

Vektörler sayı dizileridir. Bir satırı ve çok sayıda sütunu olan bir matris gibidir. Örneğin, (1, 2, 6, 4, 7, 0, 1), 7 boyutunda bir vektördür. Öznitelik vektörü, yalnızca belirli bir nesnenin özelliklerinden oluşan bir vektördür. Düğümler ve kenarlar bir ağ oluşturur. Facebook gibi bir sosyal ağı tanımlayan bir ağın, bireysel profiller olarak düğümleri ve "arkadaşlık" ilişkileri olarak kenarları vardır. Böyle bir ağın düğümleri, kişilerin adları, profil resimleri, şehirleri, ülkeleri, okul bilgileri, meslekleri, beğendikleri içerikler, yorumları, yükledikleri medya ve daha pek çok bilgiyi içeren nesnelerdir. Tüm bu veriler, bir ağdaki bir düğüm ile ilişkilendirilir ve o düğümün bir özelliği veya temsil ettiği kişinin özellikleri olarak görülebilir. Bu tür bilgiler en iyi şekilde bir öznitelik vektörü kullanılarak temsil edilir (Chami *et al.* 2022). Örneğin, Eren'in Facebook hesabı olan bir kişinin adı, şehri, üniversitesi ve ülkesi gibi bilgileri olacaktır. Bu bilgiler Eren için ("Eren SALİH", "Kerkük", "Çankırı Karatekin Üniversitesi", "Irak") olarak temsil edilebilir. Ağdaki diğer bireyler de benzer şekilde temsil edilebilir. Nesneleri öznitelik vektörleri olarak temsil etmek, gereksiz özellikleri ortadan kaldırarak tek tip bir bilgi gösterimi formatı tanımlamamıza yardımcı olur. Bu öznitelik vektörleri daha sonra analiz ve bilgi

çıkarma için çeşitli uygulamalarda veya bilgi işleme sisteminde kullanılabilir. Bir nesne için öznitelik vektörünün içeriği, büyük ölçüde nesnede bulunan veri türüne veya vektörün kullanılacağı göreve bağlıdır. Örneğin, kenarları makalelerin alıntılarını temsil eden düğümler olarak araştırma makalelerinden oluşan bir ağ, bu makalelerin her birinde en sık geçen 1000 kelimeden oluşan bir öznitelik vektörüne sahip olabilir. Örneğin, Türkçe dilinde yaygın olarak geçen sözcükleri göz ardı ederek öznitelik vektörleriyle temsil edilen aşağıdaki makaleleri göz önünde bulundurursak:

1. Makale : ("gözetimsiz", "biyolojik", " gözetimli", " yapay sinir ağlar", "analiz")
2. Makale : ("algoritma", " gözetimli", "derin öğrenme", "keras", "katman")
3. Makale : ("makine öğrenmesi", " gözetimli", "analiz", " algoritma", " yapay sinir ağlar")

Yapay zeka hakkında yazılan bir grup makaleden seçilen yukarıdaki üç makale, terim sıklığı ve ters belge sıklığı (Term Frequency - Inverse Document Frequency TF-IDF) algoritması kullanılarak çıkarılan en sık geçen kelimelerin ilgili setine sahip olabilir. Bu vektörler bize makalelerin içeriğine ilişkin üst düzey bir genel bakış sağlar ve konu dışı sözcükleri, noktalama işaretlerini, özel karakterleri vb. gibi gereksiz bilgileri ortadan kaldırır. Yukarıdaki öznitelik vektörleri sayesinde bu yayınların yapay zeka ile ilgili olduğuna karar vermemizi sağlar. Bu nedenle, öznitelik vektörleri, bir ağda bulunan bilgileri temsil etmenin ve bunlardan yararlanmanın güçlü bir yoludur. Ağ nesnelerinin belirgin özelliklere sahip olmadığı durumlarda, düğümlerin iç bağlantı sayısı, dış bağlantı sayısı, merkezilik vb. gibi yapısal öznitelikleri, ağ analizi için nesnelerin özelliklerini tanımlamak için kullanılabilir. Düğüm öznitelikleri, ağ yapısına ek olarak düğümler arasındaki içerik düzeyinde benzerliği ölçmek için doğrudan kanıt sağlayabilir. (Yang *et al.* 2015) çalışmasında gösterildiği gibi, düğüm öznitelikleri ve ağ yapısı, gürültülü bilgileri filtrelemek ve ilgili düğüm temsillerini öğrenmek için birbirlerine yardımcı olabilir.

3.3 Benzerlik

Benzerlik ölçüsü, x ve y olmak üzere iki nesne arasındaki benzerliğe sayısal bir niceleyici veren bir $B(x, y)$ işlevidir. İki nesne ne kadar benzerse, $B(x, y)$ o kadar büyük olacaktır. Çoğu benzerlik ölçüsü belirli özellikler sergiler. Örneğin, benzerlik ölçüleri genellikle (0, 1) veya bazı durumlarda (-1, 1) aralığındadır. Değerleri bu aralıklarda tutmak için ölçümleri normalleştirmemiz gerekebilir. Değerlerin bu aralıklarda olmasının nedeni, sonuçların farklı benzerlik ölçüleri üzerinden karşılaştırılmasına izin vermesidir. Bu ölçümlerin bir başka özelliği de, bir nesnenin azami ölçüde kendisine benzer olmasıdır. Örneğin, bir B benzerlik ölçüsü için, $x = y$ ise, $B(x, y) = 1$ olacaktır. Üçüncü bir özellik ise, benzerlik ölçümlerinin simetrik $B(x, y) = B(y, x)$ olması gerektiğidir (Choi *et al.* 2010).

Benzerlik ölçüleri genellikle bir mesafe ölçüsüne $M(x, y)$ dönüştürülebilir ve bunun tersi de geçerlidir. Başka bir deyişle, bir mesafe ölçüsü, bir benzerlik ölçüsünün tersi olarak görülebilir. x ve y vektörleri arasındaki mesafe Denklem (3.1)'de tanımlanan Öklid mesafe ölçüsü ile hesaplanabilir:

$$M(x, y) = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2} \tag{3.1}$$

Mesafe ölçüsünü benzerlik ölçüsüne dönüştürmek için önce Denklem (3.2)'yi kullanarak M'yi (0 - 1)'e normalleştirmemiz gerekir. Daha sonra benzerlik ölçüsü Denklem (3.3)'teki gibi hesaplanır:

$$M_norm = M \,/\, \max(M) \tag{3.2}$$

$$B = 1 - M_norm \tag{3.3}$$

Ağ verileri için üç farklı veri türü üzerinde çalışan ölçümler mevcuttur: Ağlar, kenarlar (bağlantılar) ve düğüm noktaları. Ağ benzerlik ölçüsü $B_g(G_1, G_2)$ tüm iki ağı karşılaştırır, kenar benzerlik ölçüsü $B_e((v_1, v_2), (v_3, v_4))$ iki bağlantı çiftini karşılaştırır ve düğüm

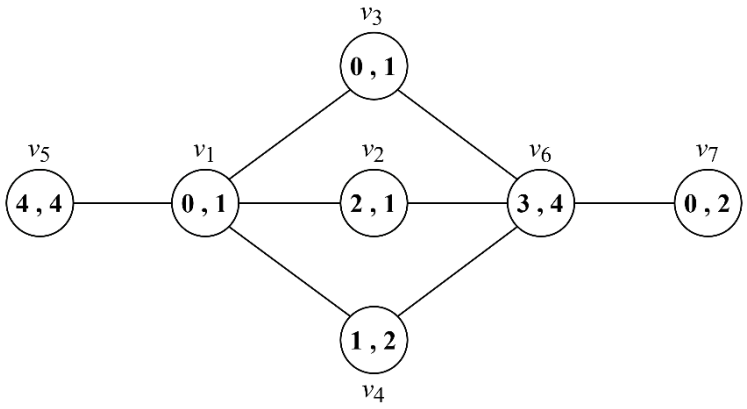
noktası benzerlik $B_v(v_1, v_2)$ ölçüsü ise, iki düğüm arasındaki benzerliği ölçer. Bu çalışmada, yalnızca düğümler arasındaki benzerlik ölçülerini kullanacağız.

Gerçek dünya ağlarında, düğümler arasında *yapısal benzerlik, anlamsal benzerlik* ve *yakınlık* olmak üzere üç farklı benzerlik türleri vardır (Kumar and Hanot 2021).

- Yapısal benzerlik açıdan bakıldığında, benzerlik yalnızca ağın yapısına bağlıdır. Bu tür düğüm yapılarının örnekleri, merkezler (hubs – çok bağlantısı olan) ve sıradışı (yalnız – çok az bağlantısı olan) düğümlerdir. Örnek olarak, bir sosyal ağdaki iki kişi, her ikisi de merkez düğüm (örneğin, fenomen) ise, yapısal olarak benzer kabul edilir (Clauset *et al.* 2004).
- Anlamsal açıdan benzerlik, yalnızca düğümlerin ve kenarların niteliklerine bağlıdır. Örneğin, tek özelliğin her bireyin yaşı olduğu sosyal ağlar alanından bir örnek alınabilir. İki kişi, sahip olabilecekleri arkadaş sayısından bağımsız olarak, aynı yaştalarsa, anlamsal olarak azami derecede benzer kabul edilir (Henderson *et al.* 2012).
- Son olarak, yakınlık açısından bakıldığında, yakın komşu düğümlerin daha fazla benzerlik paylaştığı açıktır. Dolayısıyla, bir ağdaki iki kişi, eğer yakından ilişkiliyse ve birçok komşu düğümü paylaşıyorlarsa, yakınlık açısından benzer kabul edilir (Clauset *et al.* 2004, Kumar and Hanot 2021).

Bu kavramlar, benzerlik için bir tür temel tanımlarken, bazen farklı ayarları birleştiren bir ölçümle karşı karşıya kalınabilir (Choi *et al.* 2010). Örneğin, yapısal olarak farklı kişiler, benzer niteliklere sahip oldukları için yine de yüksek bir benzerlik değeri alabilirler. Şekil 3.2, bu tür farklı kavramları göstermektedir. Burada her düğüm iki farklı öznitelik ile etiketlenmektedir. Düğüm noktaları v_1 ve v_3, aynı niteliklere sahip oldukları için anlamsal olarak maksimum düzeyde benzerdir. Bununla birlikte, komşularının sayısı farklı olduğundan, yapısal olarak yalnızca biraz benzerler. v_1 ve v_7 düğümler, bazı nitelikleri paylaştıkları için anlamsal bir bakış açısıyla biraz benzer görülebilir. Ancak yapıları birbirine hiç benzememektedir. Düğüm v_1, yapısal olarak maksimum düzeyde v_6'ya benzemektedir. v_1 ve v_5 düğümler, yapıları ve nitelikleri birbirine hiç benzemese de yakınlık benzerliği açısından çok benzer olarak görülecektir. Bu benzerlik kavramların kombinasyonunu kullanmak, v_2'nin v_1'e en benzer düğüm noktası olmasını sağlayabilir,

çünkü bunlar yalnızca bazı nitelikleri paylaşmakla kalmaz, aynı zamanda benzer bir yapıya sahip olmaya yakındır ve yakından bağlantılıdır.



Şekil 3.2 Farklı benzerlik türlerini gösteren örnek bir ağ

Bu çalışmada 2 farklı benzerlik ölçümü (Cosine ve Jaccard) ve 1 tane mesafe ölçümü (Öklid) kullanılmıştır. Benzerlik ölçümü yöntemlerin formülleri Denklem (3.4) ve Denklem (3.5)'te verilmektedir.

i. Cosine Benzerliği:

$$\boldsymbol{cosine}\ (\boldsymbol{x},\boldsymbol{y}) = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{\|\boldsymbol{x}\|*\|\boldsymbol{y}\|} \tag{3.4}$$

Burada ($\boldsymbol{x}\cdot\boldsymbol{y}$) '$x$' ve '$y$' vektörlerinin çarpımını ve ($\|\ \boldsymbol{x}\ \|*\|\ \boldsymbol{y}\ \|$) '$x$' ve '$y$' vektörlerinin uzunluklarının çapraz çarpımını temsil etmektedir.

ii. Jaccard Benzerliği:

$$\boldsymbol{Jaccard}\ (\boldsymbol{x},\boldsymbol{y}) = \frac{|\boldsymbol{x}\cap\boldsymbol{y}|}{|\boldsymbol{x}\cup\boldsymbol{y}|} \tag{3.5}$$

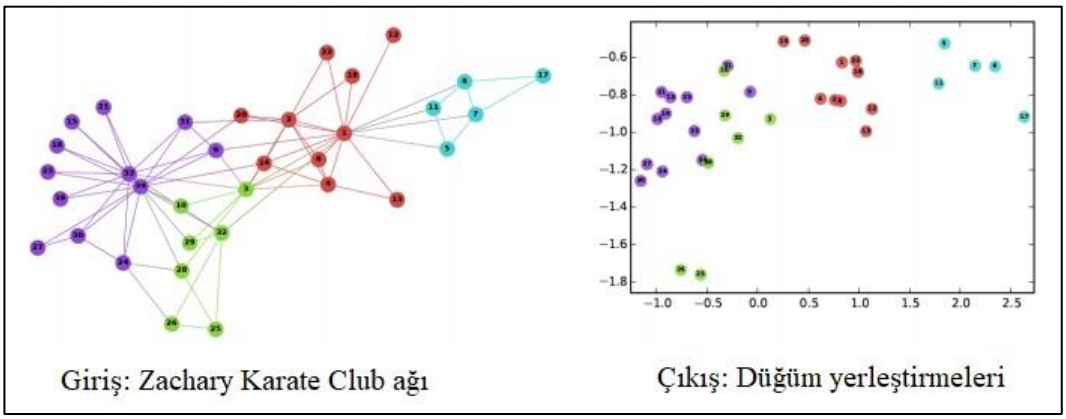
Burada ($|\boldsymbol{x}\ \cap\ \boldsymbol{y}|$) '$x$' ve '$y$' vektörlerinin kesişimdeki toplam elemanları ve ($|\boldsymbol{x}\ \cup\ \boldsymbol{y}|$) birlikteki '$x$' ve '$y$' vektörlerinin toplam elemanları temsil etmektedir.

3.4 Ağ Yerleştirme

Ağ yerleştirmenin amacı, bir ağ $G = (V, E)$ verildiğinde, $f : v \rightarrow r_v \in \mathbb{R}^d$ eşleme işlevini öğrenmektir; burada r_v , v düğümünün öğrenilmiş vektör temsilidir ve d, öğrenilen temsilin boyutudur. Burada f, orijinal ağ bilgisini korumaya çalışır, böylece orijinal ağda benzer olan iki düğüm, öğrenilen vektör uzayında da benzer şekilde temsil edilir (Zhou *et al.* 2022). Genel olarak, öğrenilen düğüm temsilleri aşağıdaki koşulları karşılamalıdır (Hamilton *et al.* 2017):

1. **Düşük boyutlu**, yani, $d \ll |V|$, başka bir deyişle, öğrenilen düğüm temsillerinin boyutu, bellek verimliliği ve sonraki ağ analiz görevlerinin ölçeklenebilirliği için orijinal komşuluk matris temsilinin boyutundan çok daha küçük olmalıdır.
2. **Bilgilendirici**, yani öğrenilen düğüm temsilleri, ağ yapısı, düğüm öznitelikleri ve düğüm etiketleri (eğer varsa) tarafından yansıtılan yakınlığı korumalıdır.

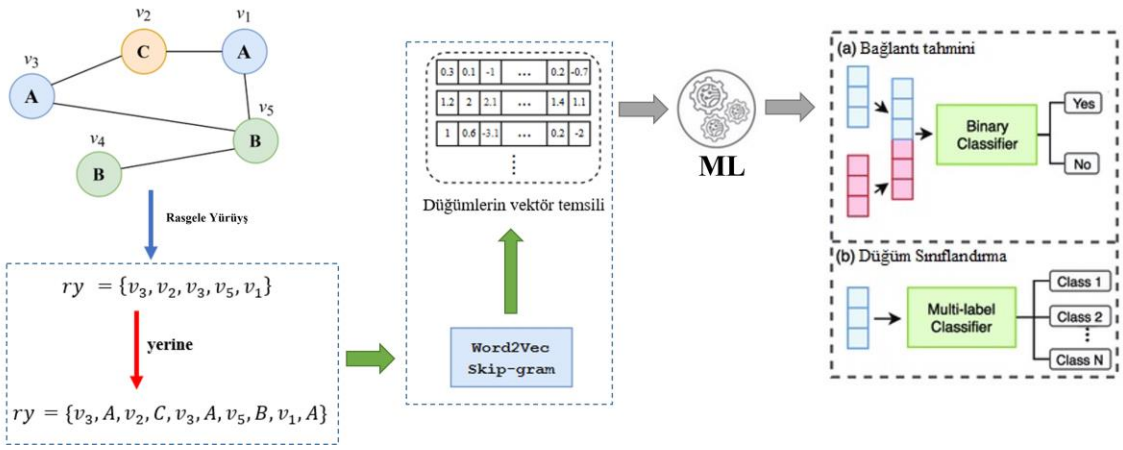
Şekil 3.3'te gösterildiği gibi, Zachary Karate Club ağı için oluşturulan düğüm yerleştirmeleri, orijinal ağdaki aynı gruba ait düğümlerin karşılık gelen yerleştirme uzayında birbirine yakın olduğu gösterilmektedir.



Şekil 3.3 Zachary Karate Club ağı için düğüm yerleştirmeleri (Perozzi *et al.* 2014)

4. MATERYAL VE METOT

Bir ağı düşük boyutlu bir uzaya yerleştirmeyi ve ağ yapısını korumak amacıyla her düğüm için vektör temsillerini öğrenmeyi amaçlayan ağ temsili öğrenimi bağlantılı verilerin işlenmesinde çok önemli bir hale gelmiştir (Zhou *et al.* 2022). Böylece, standart vektör tabanlı makine öğrenimi yöntemleri kolayca uygulanabilir. Ağ temsili öğrenmemin üzerine yapılan çalışmalar, düğüm sınıflandırma, bağlantı tahmini ve anormallik tespiti gibi analitik görevlere yardımcı olmak için ağ yapısını etkili bir şekilde yakalayabildiğini göstermiştir (Wang *et al.* 2022). Ancak, mevcut yöntemlerin çoğu yalnızca ağ yapısından yararlanır. Gerçek dünya ağları büyük miktarda nitelik bilgisi içermektedir. Bu bilgilendirici nitelikler ağ analizine fayda sağlayabilir. Dolayısıyla, bu çalışmada düğüm özniteliklerini kullanmayı mümkün kılan yeni bir yaklaşım önerilmiştir. *Node2vec-öznitelik* olarak adlandırılan bu yaklaşım, literatürde iyi bilinen ve rasgele yürüyüş istatistiğine dayanan node2vec (Grover and Leskovec 2016) algoritmasına dayanmaktadır. Önerilen yöntem hem ağın yapısal bilgisini hem de düğümlerin özniteliklerini etkin bir şekilde kodlayabilmektedir. Şekil 4.1, önerilen model tarafından kullanılan düğüm yerleştirme sürecinin görsel bir temsilini göstermektedir. Bu bölümde, node2vec ve önerilen yöntemin mimarileri detaylı bir şekilde açıklanmıştır. Ayrıca, önerilen algoritmaya topoloji ve düğüm öznitelikleri bilgilerinin birlikte nasıl entegre edildiği de bu bölümde açıklanmaktadır.



Şekil 4.1 Önerilen modelin düğüm yerleştirme süreci

4.1 Ağ Temsili Öğrenimi

Bir ağ ile ilgili bilgileri kodlamak için özelliklerin çıkarılması, geleneksel olarak, düğüm dereceleri, komşuluk matrisi ve derece matrisi gibi ağ istatistiklerine dayanan kullanıcı tanımlı buluşsal yöntemler kullanılarak yapılır. Bu teknikler, özellik çıkarımını, düğümleri ve diğer benzer işlemleri sınıflandırma görevinden önce gerçekleştirilen bir ön işleme adımı olarak ele almaktadır. Bu teknikler verimli olsa da, binlerce veya daha fazla düğüme sahip daha büyük ağlar için ciddi zorluklar getirmektedir. Örneğin, geleneksel ağ algoritmalarının çoğu, genellikle zaman ve alan konusunda yüksek hesaplama maliyetlerine neden olan komşuluk matris üzerinde çalışmaktadır (Neville and Jensen 2000, Zhu *et al.* 2003). Ağlar yüz milyonlarca, milyarlarca noktaya sahip olabilirler. Bu tür bir ağ için komşuluk matris gösterimi 10^{12} elemana ihtiyaç duyacaktır. Bu da şu anki hesaplama ve depolama kapasiteleri ile mümkün değildir (Zhou *et al.* 2022).

Ağ verileri ve diğer metriklerden oluşan elle tasarlanmış (hand-engineered) özelliklerin kullanılması zahmetli bir prosedürdür ancak kullanılabilecek alternatif bir yöntem olmuştur (Gallagher and Eliassi-Rad 2010). Pregel (Malewicz *et al.* 2010), Giraph (Martella *et al.* 2015) ve Graphx (Xin *et al.* 2013) gibi diğer ağ analitik platformları, ölçeksiz özelliğine sahip gerçek dünya ağları için sınırlı verimliliğe sahip olmuşlardır. Geleneksel tekniklerin bu eksikliğinin üstesinden gelmek için, özellik çıkarma görevi için makine öğrenimi ve boyut indirgeme ilkelerini kullanan daha yeni yöntemler geliştirilmiştir. Bu yöntemler, makine öğrenimi yöntemlerini ve doğrusal olmayan boyutluluk azaltma ilkelerini kullanarak düğümleri veya ağ yapısının tamamını d boyutlu bir vektöre kodlamayı otomatik olarak öğrenir.

Yapısal veriler için temsilleri öğrenmek (özellik çıkarma görevi), son zamanlarda büyük ilgi gören bir konu olmuştur (Hamilton *et al.* 2017, Zhou *et al.* 2020). Literatürdeki mevcut yaklaşımlar üç geniş gruba ayrılmaktadır: (i) Matris çarpanlarına ayırmayı (matrix-factorization) kullanan yöntemler, (ii) Olasılık temelli modeller ve (iii) Derin öğrenme tabanlı yöntemler.

4.1.1 Matris çarpanlarına ayırmayı (matrix-factorization) kullanan yöntemler

Matris çarpanlarına ayırma, bir matrisin matrislerin çarpımına ayrıştırılmasıdır. Bu anlamda serideki ilk matris öz düğüm temsili olarak adlandırılır ve son matris ise düğüm bağlamını ifade eder. Çarpanlara ayırma modelleri, farklı makine öğrenimi alanlarında anlamlı düşük boyutlu temsilleri elde etmek için yaygın kullanılan tekniklerdir. Bu yöntemler, boyut küçültme ve çok boyutlu ölçeklendirme kavramlarına dayanan klasik tekniklerden esinlenmektedir (Zhou *et al.* 2022). Çizge temsili (graph representation – GraRep) (Cao *et al.* 2015), yüksek sıralı yakınlık korumalı yerleştirme (high-order proximity preserved embedding - HOPE) (Ou *et al.* 2016) gibi matris çarpanlarına ayırma tabanlı algoritmalar, ağ yapısını bir komşuluk matrise önceden işler ve komşuluk matrisini ayrıştırarak düğüm yerleştirmeleri elde eder. Bir ağın yakınlık matrisini oluşturmak, boyutsallığı azaltır. Dolayısıyla, IsoMap (Tenenbaum *et al.* 2000) en kısa yol matrisi kullanır ve yerleştirmeleri öğrenmek için çok boyutlu ölçekleme (multidimensional scaling - MDS) metodunu uygular. LLE (Roweis and Saul 2000), komşu düğümlerin birbirini etkilediği ağırlık matrisini yeniden oluşturarak düğüm benzerliğini öğrenir ve elde edilen ağırlık matrisi ile düğüm yerleştirmelerini öğrenmek için bu prosedürü tekrarlar. Matris çarpanlarına ayırmanın yüksek hesaplama maliyeti nedeniyle, bu algoritmalar büyük ölçekli verileri işlemekle sınırlıdır.

4.1.2 Olasılık temelli modeller

(Perozzi *et al.* 2014, Tang *et al.* 2015, Grover and Leskovec 2016, Nguyen and Malliaros 2018, Ahmed *et al.* 2020, Mohammed and Gündüç 2023) gibi olasılıksal modeller, ağdan farklı desenleri çıkararak düğüm yerleştirmelerini öğrenmeye çalışır. Çıkartılan desenler veya yürüyüşler, global yapısal eşdeğerliği, yerel komşuluk bağlantılarını ve birinci, ikinci veya daha yüksek dereceli yakınlığı içermektedir. Matris çarpanlarına ayırma yönteminin sınırlamalarına yanıt olarak, düğümlerin yerel komşuluklarını ve özelliklerini korumaya çalışan rasgele yürüyüşlere veya yakınlık matrislerine dayalı olarak farklı yaklaşımlar ortaya çıkmıştır. Cümlelerdeki kelimelerin bağlamını kullanarak kelime temsillerini öğrenmek için geliştirilen Skip-gram (Mikolov *et al.* 2013a) modeline dayalı olarak DeepWalk (Perozzi *et al.* 2014) ve node2vec (Grover and Leskovec 2016), bir

ağdaki düğümlerin temsillerini öğrenebilir. Burada ana fikir, Mikolov ve diğerleri (Mikolov *et al.* 2013b) tarafından doğal dil işleme (Natural Language Processing - NLP) uygulamalarında başlatılan Skip-gram modeli çizgisini izleyerek, yerleştirmesi verilen bir düğümün komşuluğunu gözlemleme olasılığını en üst düzeye çıkarmaktır. DeepWalk'ta rasgele yürüyüşlerle oluşturulan kısa düğüm dizileri cümleler olarak alınır ve düğüm temsilleri, rasgele yürüyüş dizilerindeki bağlamsal bilgiler kullanılarak öğrenilir. Ancak, DeepWalk tarafsız ve kesik rasgele yürüyüşleri kullanarak yalnızca lokal komşuluk bilgisini yakalayabilir ve global topoloji bilgisini yakalamaktan eksik kalır. DeepWalk'a benzer şekilde, node2vec (Grover and Leskovec 2016) orijinal ağdan hem lokal hem de global topoloji bilgilerini dikkate alarak düğüm dizilerini örneklemek için geliştirilmiş taraflı rasgele yürüyüş yaklaşımını kullanır. Böylece ağlarda gözlemlenen bağlantı modellerinin çeşitliliğini yakalar. Node2vec belirli bir düğümün yeniden ziyaret sıklığını kontrol etmek için kullanılabilecek 2 ek parametre kullanır. İlk parametre geriye dönüş parametresi olarak adlandırılır ve p ile gösterilir. Yürüyüşte bir düğümü ne sıklıkta tekrar ziyaret ettiğimizi yönetir. (*InOut*) olarak adlandırılan ikinci parametre q ile gösterilir. Yürüyüşün başlangıç düğümünü ne kadar hızlı keşfedip terk ettiğini kontrol eder. Node2vec ve DeepWalk arasındaki fark ince ama önemlidir. DeepWalk rasgele seçilen düğümlerinin komşularını ziyaret ederken tamamen rasgele bir strateji kullanılırken, node2vec daha rehberli bir arayış içerisindedir. Node2vec arayışı, her adımın bir sonraki hareketini başka bir düğüme yönlendiren bir geçiş olasılığı üzerinden yönlendirilir. Rasgele yürüyüş tabanlı modellerin matris çarpanlarına ayırma yöntemine karşı en büyük avantajı, tüm düğüm çiftlerini eğitmesi gerekmemesi, yalnızca rasgele yürüyüşlerde birlikte oluşan düğüm çiftlerini dikkate almasıdır. Rasgele yürüyüş tabanlı yöntemlerden farklı olarak, LINE (Tang *et al.* 2015), sırasıyla 1 adımlı ve 2 adımlı düğüm ilişkilerini yakalamak için iki farklı yakınlık modeli sunar. Bir kenarı paylaşıyorlarsa iki düğüm benzer olarak kabul edilir yeni yerleştirme uzayında yakından temsil edilme eğiliminde olduğu birinci dereceden yakınlık ve birçok komşu/bitişik düğüm paylaşıyorlarsa iki düğüm benzer gizli temsillere sahip olma eğiliminde olduğu ikinci dereceden yakınlık. LINE, iki model tarafından öğrenilen düğüm temsilleri, son düğüm temsilleri olarak birleştirilir.

Yerel bağlantı modellerinin yanı sıra, düğümler genellikle merkez düğümler (*hub*) veya klik üyeleri gibi benzer yapısal rolleri paylaşır. Yapısal rol yakınlığını koruyan ağ temsilleri, birbirinden uzakta olan ancak benzer yapısal rolleri yakından paylaşan ağ düğümlerini yerleştirmeyi amaçlar. Bu, yalnızca yapısal rol bağımlı görevleri kolaylaştırmakla kalmaz, aynı zamanda yerel yapıyı koruyarak ağ yerleştirmeyi de geliştirir. Struct2vec (Ribeiro *et al.* 2017), ilk olarak düğüm yapısal rolü benzerliğini çok katmanlı bir ağa kodlar; burada her katmandaki kenarların ağırlıkları karşılık gelen ölçekteki yapısal rol farkı tarafından belirlenir. DeepWalk (Perozzi *et al.* 2014) daha sonra düğüm temsillerini öğrenmek için çok katmanlı ağ üzerinde gerçekleştirilir. Dolayısıyla, çok katmanlı ağda birbirine yakın düğümler (yüksek yapısal rol benzerliği ile) yeni temsil uzayına yakından yerleştirilir.

Sözü geçen tüm algoritmalar, düğüm temsillerini denetimsiz bir şekilde öğrenir. Öğrenilen düğüm temsillerinin, hedeflenen düğüm sınıflandırma görevine fayda sağlamak amacıyla düğüm etiketleriyle ağ temsilleri geliştirmek için bazı araştırma çabaları yapılmıştır. Max-Margin DeepWalk (MMDW) (Tu *et al.* 2016), DeepWalk'ın matris çarpanlarına ayırma versiyonunun (Yang *et al.* 2015) hedefini çok sınıflı destek vektör makinesi (SVM) hedefi ile birleştirir. Transdüktif büyük ölçekli bilgi ağı yerleştirme (transductive large-scale information network embedding - TLINE) (Zhang *et al.* 2016b) ise, LINE yönteminin (Tang *et al.* 2015) düğüm temsillerini ve bir SVM sınıflandırıcısını aynı anda öğrenen LINE yönteminin yarı denetimli bir uzantısı olarak önerilmiştir. Düğümlerin, benzer komşuları paylaşmaları veya benzer gruplara katılmaları durumunda düşük boyutlu uzaya yakın bir şekilde yerleştirmeleri gerektiğini varsayarsak, grup gelişmiş ağ yerleştirme (group enhanced network embedding - GENE) (Chen *et al.* 2016) grup (etiket) bilgisini ağ yapısıyla olasılıksal bir şekilde birleştirir.

4.1.3 Derin öğrenme tabanlı yöntemler

Derin öğrenme yaklaşımları (Gao *et al.* 2018), özellikle otomatik kodlayıcı tabanlı yöntemler de ağ yapısından düğüm temsillerini öğrenmek için geniş çapta kullanılmaktadır. Ağ temsilleri için derin sinir ağları (deep neural networks for graph representations – DNGR) (Cao *et al.* 2016) rasgele yürüyüş yöntemiyle ilk önce düğüm

temsillerini koruyan yüksek boyutlu vektör elde eder ve ardından düşük boyutlu temsilleri öğrenmek için yığılmış gürültü giderici otomatik kodlayıcıyı (stacked denoising autoencoder - SDAE) (Vincent *et al.* 2010) kullanır. Wang ve arkadşları (Wang *et al.* 2016) yapısal derin ağ yerleştirme (structural deep network embedding - SDNE) adlı yarı denetimli derin bir model önermişlerdir. Ağ yapısının hem lokal hem de global doğru yapı tanımlarını korumak için hem birinci hem de ikinci dereceden yakınlıkları optimize etmeyi önermişlerdir. SDNE, komşuluk matrisini derin bir otomatik kodlayıcıya verir ve ardından, θ parametre kümesine (yani, ağırlıklar ve eğilimler) göre bir kayıp fonksiyonunu L_{mix} en aza indirerek yeniden oluşturma hatasını optimize eder. SDNE modeli, çeşitli ağ verilerinde düğüm sınıflandırma, bağlantı tahmini ve görselleştirme görevlerine uygulanıp diğer algoritmalardan daha iyi performans göstermektedir. GraphGAN (Wang *et al.* 2019), iki bileşenli çekişmeli bir öğrenme platform aracılığıyla düğüm bağlantı davranışını modelleyerek düğüm temsillerini öğrenir: (i) v_c'ye bağlı düğümlerin dağılımına uyan ve muhtemel bağlı düğümleri oluşturan üretici (Generator - $G(v|v_c)$) ve (ii) $G(v|v_c)$ tarafından üretilen düğüm çiftlerini gerçek olanlardan ayırt etmek için düğüm çifti $(v|v_c)$ için bir bağlantı olasılığı veren ayırıcı (Discriminator - $D(v|v_c)$). GraphGAN, üretici ve ayrımcı arasındaki rekabetten yararlanarak düğüm bağlantı davranışına dikkat eden bilgilendirici düğüm temsillerini öğrenir. Literatürde diğer birkaç çalışma, ağlar için belirli makine öğrenimi görevlerine derin öğrenme mimarileri uygulamaktadır. Li *et al.* (2014) çalışmalarında, dinamik bağlantı tahmini için koşullu geçici kısıtlı Boltzmann makinesi (conditional temporal restricted Boltzmann machine - ctRBM) modelini önermektedir (Li *et al.* 2014). Derin mimariler ayrıca heterojen ağlardaki düğüm temsillerini öğrenmek için de kullanılmıştır (Wang *et al.* 2022).

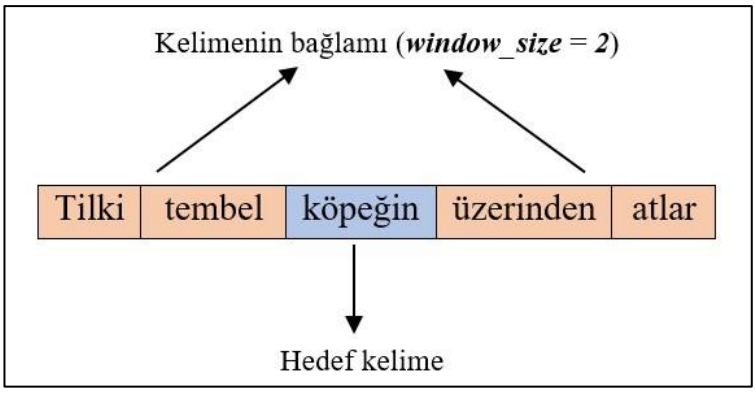
4.2 Kelime Yerleştirmeleri

Kelime yerleştirmeleri, her kelimenin benzersiz bir gerçek değerli vektöre eşlendiği, dolayısıyla benzer kelimelerin benzer yerleştirme vektörlerine sahip olduğu kelimelerin vektör temsilleridir (Bengio *et al.* 2000, Collobert and Weston 2008, Mikolov *et al.* 2013b, a, Pennington *et al.* 2014). Duygu analizi (Socher *et al.* 2013), soru yanıtlama (Weston *et al.* 2016), konu modelleme (Nguyen *et al.* 2015), makine çevirisi (Zou *et al.*

2013) ve dizi etiketleme (Ma and Hovy 2016) gibi birçok başarılı NLP uygulamasında kelime yerleştirme modelleri önemli bir katkı sağlamıştır.

Word2vec , kelimeleri vektör uzayında ifade etmeye çalışan gözetimsiz (*unsupervised*) ve tahmin temelli (prediction-based) bir modeldir. Google araştırmacı Tomas Mikolov ve ekibi tarafından 2013 yılında icat edilmiştir (Mikolov *et al.* 2013b, a). 2 çeşit alt yöntemi vardır: Sürekli sözcük torbası (continous bag of words - CBOW) ve Skip-gram. İki yöntem de genel olarak birbirine benzemektedir.

Word2vec için en önemli hiperparametrlerden bir tanesi *window_size*'dir. Window_size hiperparametresi bize ortadaki kelimenin sağında ve solunda kaç kelime olabileceğini ifade ediyor. Şekil 4.2'de window_size'ın 2 olduğu bir örnek verilmektedir.



Şekil 4.2 window_size'ın 2 olduğu bir örnek

CBOW ve Skip-gram modelleri birbirlerinden çıktı'yı (output) ve girdi'yi (input) alma açısından farklılaşmaktadır. CBOW modelinde *window size*'ın merkezinde olmayan kelimeler girdi olarak alınıp, merkezinde olan kelimeler çıktı olarak tahmin edilmeye çalışırken; Skip-gram modelinde ise merkezdeki kelime girdi olarak alınıp merkezde olmayan kelimeler çıktı olarak tahmin edilmeye çalışılır. Bu işlem cümle bitene kadar devam eder. Bir cümleye uygulanan bu işlemler tüm cümlelere uygulanır ve böylece başlangıçta elimizde bulunan etiketsiz veriye mapping işlemi uygulanmış olur ve eğitim etmeye hazır olur (Grohe 2020).

W kelime dağarcığında w kelime türünün "hedef" vektör temsili v_w ile gösterilirken, "bağlam" (komşu kelimler) vektör temsili $\tilde{v}_w$ ile gösterilir. $(w_1, w_2, \ldots, w_T)$ kelime dizisi verildiğinde, Skip-gram log-olasılık (log-likelihood) amaç fonksiyonunu Denklem (4.1)'deki gibi maksimize eder (Mikolov *et al.* 2013b):

$$\sum_{t=1}^{T} \log P(w_{t-c}, \ldots w_{t-1}, w_{t+1}, \ldots, w_{t+c} \mid w_t) \quad (4.1)$$

Burada T, Skip-gram modelini eğitmek için oluşan bir dizi cümle kümesidir (corpus) ve c, w_t hedef kelimesinin önceki ve sonraki kelime sayısıdır (window size). Büyük c boyutları, eğitim süresi pahasına daha doğru tahminler sağlayabilir.

$P(w_{t-c}, \ldots w_{t-1}, w_{t+1}, \ldots, w_{t+c} \mid w_t)$ olasılığı Denklem (4.2)'de gösterildiği gibi hesaplanır.

$$\sum_{-c \le j \le c, j \neq 0} \log P(w_{t+j} \mid w_t) \quad (4.2)$$

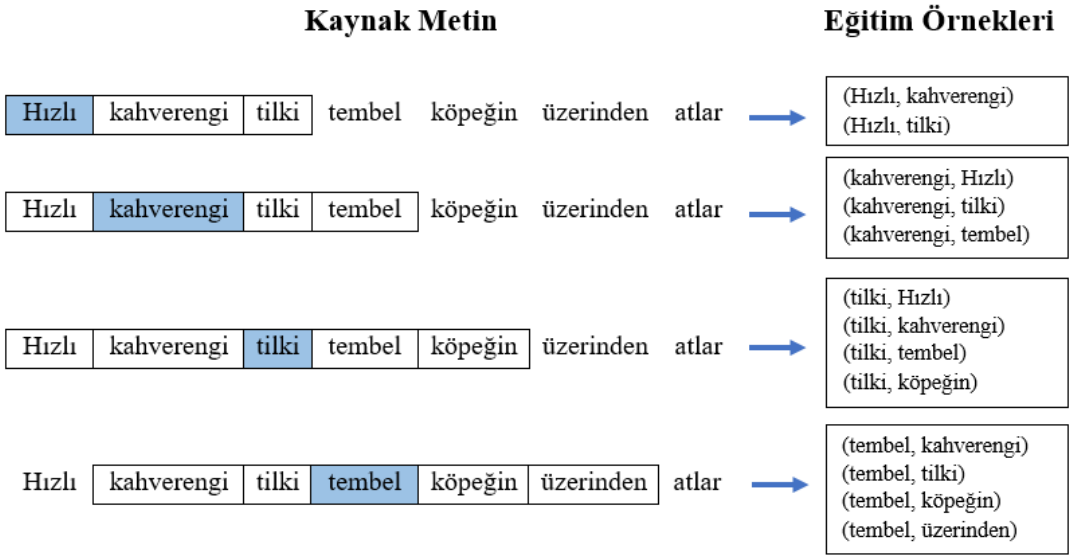
Burada w_{t+j}, w_t'nin bağlam sözcüklerinden biridir. $P(w_{t+j} \mid w_t)$ olasılığı genellikle Denklem (4.3)'te tanımlandığı gibi softmax işlevi kullanılarak tanımlanır.

$$P(w_o \mid w_i) = \frac{\exp\left(\tilde{v}_{w_o}^{\top} v_{w_i}\right)}{\sum_{w=1}^{W} \exp\left(\tilde{v}_{w}^{\top} v_{w_i}\right)} \quad (4.3)$$

Burada w_o bağlam kelimesine ve w_i hedef kelimeye karşılık gelir. Skip-gram modelinin nihai çıktı olasılıkları, W kelime dağarcığında her kelime teriminin giriş kelimesine ne kadar yakın olduğunu gösterecektir. Örneğin, eğitimli Skip-gram modeline "Sovyet" girdi kelimesi verildiyse, "Birlik" ve "Rusya" gibi kelimelerin çıktı olasılıkları, "portakal" ve "panda" gibi ilgisiz kelimelere göre çok daha yüksek olacaktır.

Skip-gram modeline eğitim belgelerinde bulunan kelime çiftlerini besleyerek eğitim gerçekleşecektir. Şekil 4.3, "Hızlı kahverengi tilki tembel köpeğin üzerinden atlar" cümlesinden oluşturulacak bazı eğitim örneklerini (kelime çiftlerini) göstermektedir.

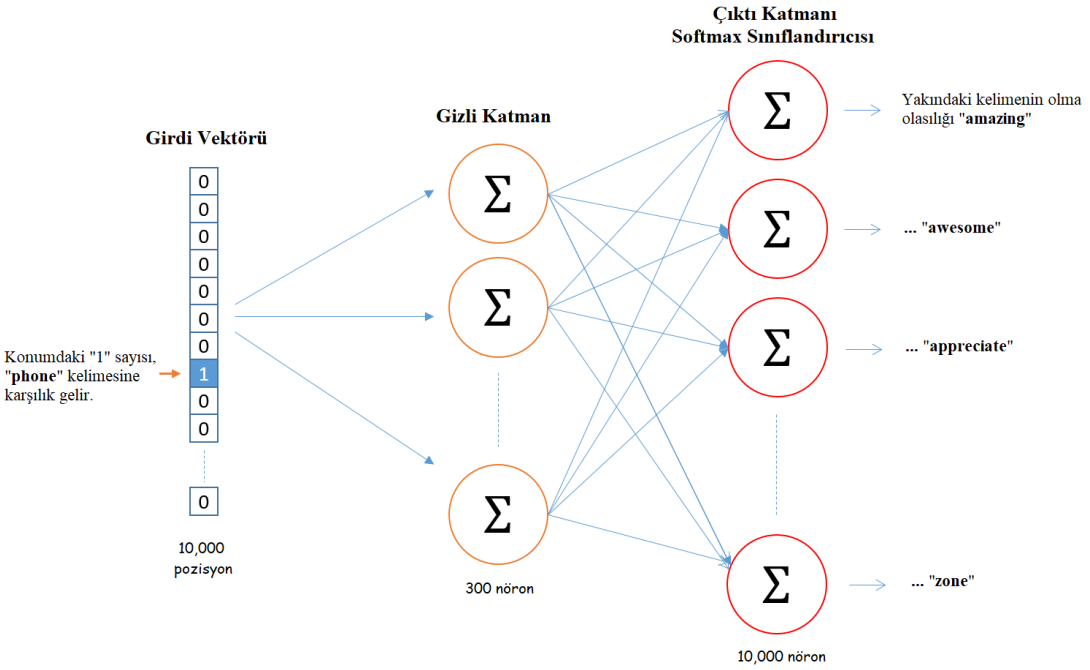
Burada pencere boyutu (window size) 2 kullanılmıştır. Mavi ile vurgulanan kelime giriş kelimesidir. Skip-gram modeli istatistikleri her bir eşleşmenin kaç kez göründüğünden öğrenecektir. Örneğin, model muhtemelen ("Sovyet", "Birlik") kelime çiftini eğitim örneğinden ("Sovyet", "portakal") kelime çiftinden çok daha fazla alacaktır. Eğitim tamamlandığında, modele girdi olarak "Sovyet" kelimesini verdiğimizde, "Birlik" veya "Rusya" için, "portakal" olasılığından çok daha yüksek bir olasılık verecektir.



Şekil 4.3 Skip-gram modeli için oluşturulan veri örneği

Skip-gram modelinin mimarisi Şekil 4.4'te gösterilmektedir. Burada 10000 benzersiz kelimeden oluşan corpus'a gizli katman ile çıkış katman arasına softmax uygulanarak olasılıklar hesaplanmaktadır. Skip-gram yönteminin sinir ağı, bir kelimeyi girdi olarak alır ve bağlam kelimelerini tahmin etmeye çalışır. İlk olarak, bir sinir ağına metin dizilerini veremez, bu nedenle kelimeler temsil edilmelidir. Önce eğitim verimizden örneğin 10000 benzersiz kelimeden oluşan bir kelime dağarcığı oluşturulur ($W = 10000$). Sonra "phone" gibi bir giriş kelimesi one-hot vektör olarak temsil edilir. Bu vektör ($1 \times W$) 10000 bileşene sahip olacak (kelime dağarcığımızdaki her kelime için bir tane) ve "phone" kelimesinin karşılık geldiği yere "1", diğer tüm konumlara 0 atanır. Skip-gram sinir ağı, tek bir gizli katmandan oluşur. Gizli katman nöronlarında aktivasyon fonksiyonu yoktur, ancak çıkış nöronları softmax aktivasyon fonksiyonu kullanır. Tek gizli katman, $W \times d$ boyutuna sahip olacaktır; burada d, yerleştirilen kelimenin

boyutudur ve bir hiper parametredir. Gizli katmanın çıktısı, bir softmax katmanına besleyeceğimiz $1 \times d$ boyutunda olacaktır. Çıktı katmanının boyutları $1 \times W$ olacaktır ve vektördeki her değer, o konumdaki hedef kelimenin olasılık puanı olacaktır. Kısaca Skip-gram sinir ağı istatistikleri her bir eşleşmenin kaç kez göründüğünden öğrenecektir.



Şekil 4.4 Skip-gram modelinin mimarisi

CBOW modeli ise, çevredeki kelimelerin bağlamını anlamaya çalışarak hedef kelimeyi tahmin etmeye çalışır. CBOW, hedef kelimeyi tahmin etmek için bağlam kelimelerinin vektör temsillerinin ortalamasını $\tilde{v}_{avg}$ vektörüne alır. CBOW modeli log-olasılık (log-likelihood) amaç fonksiyonunu Denklem (4.4), Denklem (4.5) ve Denklem (4.6) ile maksimize eder (Mikolov *et al.* 2013b):

$$= \sum_{t=1}^{T} \log \mathrm{P}(w_t \mid w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}) \tag{4.4}$$

$$= \sum_{t=1}^{T} \log \mathrm{P}(w_t \mid \mathrm{avg}) \tag{4.5}$$

$$= \sum_{t=1}^{T} \log \frac{\exp(v_{w_t}^{\top} \tilde{v}_{\mathrm{avg}})}{\sum_{w=1}^{W} \exp(v_{w}^{\top} \tilde{v}_{\mathrm{avg}})} \tag{4.6}$$

Mikolov ve çalışma arkadaşlarına göre (Mikolov *et al.* 2013b), CBOW modelleri genel olarak küçük veri setlerde daha iyi çalışırken, büyük veri setlerde Skip-gram daha iyi çalışmaktadır. CBOW daha az hesaplama gücü gerektirirken, Skip-gram daha fazla hesaplama gücü gerektirir. CBOW 2 veya daha çok anlamlı kelimeleri anlamakta iyi değilken Skip-gram 2 veya daha çok anlamlı kelimeleri daha iyi öğrenebilmektedir. Ancak Skip-gram modeli, seyrek kelimeleri tahmin etme yeteneği nedeniyle çoğu zaman daha iyi bir seçim olmuştur.

4.3 Node2vec

Grover ve Leskovec (2016) tarafından geliştirilen node2vec, ağlarda düğüm yerleştirme öğrenimi için verimli ve ölçeklenebilir bir yöntemdir. Palumbo vd. (2017) node2vec'i rastgele yürüyüşlerle örneklenen düğüm dizileri üzerine word2vec modelinin uygulanmasıyla bir ağdaki düğümlerin gösterimini öğrenen bir algoritma olarak tanımlamaktadır (Palumbo *et al.* 2017). Node2vec, word2vec kelime yerleştirme işleminde kullanılan Skip-gram modelini kullanır. Amaç, belirli bir hedef düğüm için komşu düğümleri tahmin etmektir. Node2vec yöntemi düğümlerin ağ komşuluklarını koruma olasılığını en üst düzeye çıkararak, düğümlerin düşük boyutlu Öklid uzayına eşlenmesini gerçekleştirir. Denklem (4.7)'de node2vec için amaç fonksiyonu tanımlanmaktadır.

$$\max_{f} \sum_{u \in V} \log P(N_S(u) \mid f(u)) \tag{4.7}$$

Burada u bir düğüm noktasıdır, örnekleme stratejisi (S) ile edilen u düğümünün ağ komşuluğu $N_S(u)$ olarak gösterilir. $f(u)$ ise u düğüm noktasının yerleştirmesini temsil eder. Bu amaç fonksiyonu, Skip-gram amaç fonksiyonuna çok benzer, burada $f(u)$ giriş sözcüğüdür ve $N_S(u)$ bağlamdır, dolayısıyla bu işlevi çözmek için yazarlar Skip-gram yaklaşımını kullanmaktadırlar (Grohe 2020). Skip-gram modeli, node2vec algoritmasının önemli bir bileşenidir. Esas olarak veriler özellik vektörleri oluşturmak için kullanılır. Ancak, Skip-gram uygulanmadan önce çözülmesi gereken bir problem vardır, o da hedef

düğümün lokal komşuluk bilgilerinin nasıl temsil edilmesi gerektiğinin belirlenmesidir (Grover and Leskovec 2016, Goyal and Ferrara 2018).

Node2vec, yanlı (biased) rasgele yürüyüş sunarak DeepWalk (Perozzi *et al.* 2014) ile tanıtılan yansız (unbiased) rasgele yürüyüş kullanımını daha da ileri götürmeye çalışır. Bu yürüyüşler sayesinde cümle benzeri yapılar (düğüm dizileri) elde eder. Node2vec, genişlik ilk arama (BFS) ve derinlik ilk arama (DFS) olmak üzere iki örnekleme stratejisinden faydalanır, böylece ağlarda gözlenen bağlantı modellerinin çeşitliliğini yakalar. BFS, kaynak düğümün yakın çevresini örnekler ve DFS, kaynak düğümden giderek uzaklaşan düğümleri örnekler (Grover and Leskovec 2016, Grohe 2020).

Node2vec algoritmasının örnekleme stratejisi, eğitim kümesini oluşturmak için bir karar verisi oluşturmak amacıyla belirli bir düğümün komşusundan öğeleri seçmesidir (Grover and Leskovec 2016). Node2vec'in örnekleme stratejisi 4 adet parametre kabul eder, bunlar:

- Düğüm başına yürüyüş sayısı (nw): ağdaki her düğümden oluşturulacak rastgele yürüyüşlerin sayısıdır.
- Rasgele yürüyüş uzunluğu (wl): her rastgele yürüyüşte geçilen düğüm sayısıdır.
- p : önceden ziyaret edilmiş bir düğüme geri dönme olasılığını kontrol eden dönüş hiper parametresi olarak bilinir.
- q : başlangıç düğümünden uzaklaşma olasılığını kontrol eden hiper parametresidir.

Node2vec, örnekleme stratejilerini doğrudan kullanmaz ancak bir düğüm noktasının komşuluğunu (lokal çevresini) temsil etmek için yanlı rasgele yürüyüşler kullanır. Yürüyüşler, p ve q parametreleriyle kontrol edilir. p parametresi, yürüyüşte düğümleri tekrar ziyaret etme olasılığını etkiler. Yüksek bir p değeri ($> \max(q, 1)$) daha önceden ziyaret edilmiş düğümlerin yeniden ziyaret edilme olasılığının düşük olmasını sağlar, bu da DFS benzeri yani ağın makul bir şekilde keşfedilmesini teşvik eder. Düşük bir p değeri ($< \min(q, 1)$) ise rasgele yürüyüşü bir adım geri izlemeye yönlendirir ve BFS'nin yaptığı gibi yürüyüşü yerel olarak hedef düğüm noktasının yakınında tutar. q parametresi ise,

yürüyüşün başlangıç düğüm noktasına yakın veya uzak olan ziyaret düğümlere doğru eğilimli olup olmadığını etkiler. $q > 1$ ise yürüyüş, hedef düğüm noktasına yakın düğümlere doğru eğilimlidir ve BFS davranışına benzer bir davranış sergiler. Eğer $q < 1$ ise, yürüyüş, DFS benzeri bir keşfi yansıtan, hedef düğüm noktasından uzaktaki ziyaret düğümlerine doğru eğilimli olur (Grover and Leskovec 2016, Grohe 2020, Salamat *et al.* 2020).

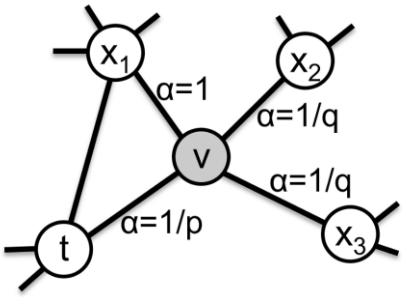
Yanlı rasgele yürüyüşün normalleştirilmemiş geçiş olasılığı Denklem (4.8)'deki gibi tanımlanır:

$$\pi_{vx} = \alpha_{pq}(t, x) \cdot w_{vx} \tag{4.8}$$

Burada $v, x, t \in V$, v geçerli düğüm noktasıdır, t önceki düğüm noktasıdır ve x sonraki düğüm noktasıdır. Burada $\alpha_{pq}(t, x)$, Denklem (4.9)'da görüldüğü gibi p ve q tarafından tanımlanan yanlı aramadır ve w_{vx} ise (v, x) düğüm çifti arasındaki bağlantı ağırlık değeridir, ağırlıksız ağlar için $w_{vx} = 1$.

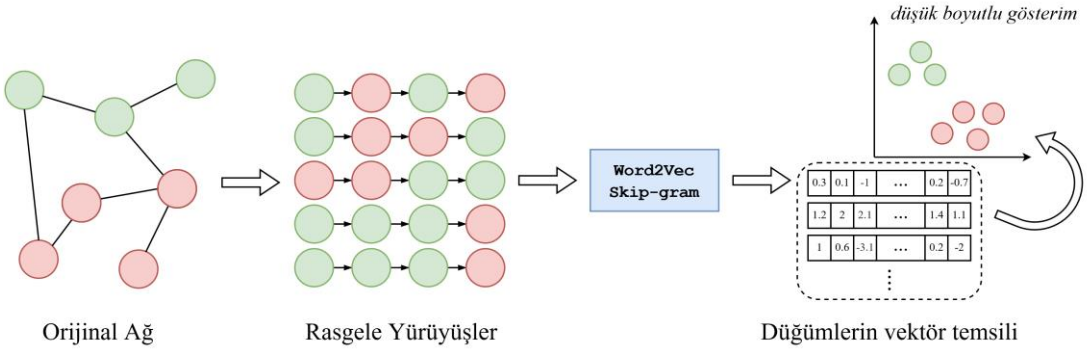
$$\alpha_{pq}(t, x) = \begin{cases} \frac{1}{p} & \text{eğer} \quad d_{tx} = 0 \\ 1 & \text{eğer} \quad d_{tx} = 1 \\ \frac{1}{q} & \text{eğer} \quad d_{tx} = 2 \end{cases} \tag{4.9}$$

Burada d_{tx}, t düğümünden x düğümüne olan en kısa yol mesafesidir. Bunun bir örneği Şekil 4.5'te gösterilmektedir, burada yürüyüş t düğüm noktasından başlayıp ve şimdi v düğüm noktasında durmaktadır. Şekil 4.5'te p'nin düşük değerlerinin yürüyüşü geriye doğru teşvik ettiği ve q'nun düşük değerlerinin hedef düğüm noktasından uzaklaşmaya yol açtığı görülebilir. Her hedef düğümün α değeri, ziyaret edilen önceki düğüm olan t'ye olan uzaklaığına göre hesaplanır (Grover and Leskovec 2016, Zhou *et al.* 2022).



Şekil 4.5 Node2vec algoritmasında geçiş olasılığının nasıl hesaplanması (Grover and Leskovec 2016)

Node2vec, ön işleme ile başlayan, yürüme simülasyonları ve optimizasyon ile biten üç adımlı bir tekniktir. Geçiş olasılıkları, p ve q parametrelerine dayalı olarak ön işleme aşamasında hesaplanır. Yürüme simülasyonları aşamasında, her düğüm noktası için nw kez yanlı rasgele yürüyüşler oluşturulur. Denklem (4.7)'deki hedef fonksiyonu, optimizasyon aşamasında maksimize edilir. Maksimize etmek için, node2vec stokastik gradyan iniş yaklaşımını kullanır ve nihai sonuç f, her düğüm için $1 \times d$ boyutunda bir yerleştirme (vektör temsili) olacaktır (Goyal and Ferrara 2018, Zhou *et al.* 2022).



Şekil 4.6 Node2vec modelin ardışık düzeni

Herhangi bir ağ verildiğinde, düğüm yerleştirmelerini oluşturmak için node2vec modelin ardışık düzeni Şekil 4.6'te gösterilmektedir. Önce belirli bir ağdan rasgele yürüyüş dizileri oluşturur ve ardından düğüm temsillerini öğrenmek için Skip-gram modeli uygulanır. Tüm düğümler için öğrenilen düğüm yerleştirme özellikleri, bağlantı tahmini, düğüm sınıflandırması ve topluluk tespiti gibi farklı ağ analiz görevleri için kolayca ve verimli bir şekilde kullanılabilir.

Gerçek dünya ağları büyük miktarda öznitelik bilgisi içermektedir. Ancak node2vec yöntemi düğüm öznitelikleri gibi bu tür değerli bilgileri kullanmamaktadır yalnızca ağ yapısından yararlanır. Bir sonraki bölümde, node2vec'in bu ek bilgileri kullanabilecek şekilde nasıl genişletildiği detaylı açıklanmıştır.

4.4 Önerilen Model

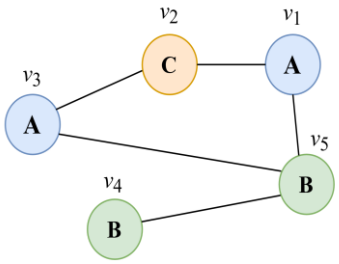
Node2vec algoritmasında (Grover and Leskovec 2016) rasgele bir yürüyüş (ry), bir düğümler dizisi olarak Denklem (4.10)'daki gibi temsil edilir.

$$ry = \{v_1, v_2, v_3, \dots, v_n\} \tag{4.10}$$

burada her düğüm (v) ayrı bir tanımlayıcı (identifier - id) ile gösterilir. Gerçek dünya ağlarının öznitelikleri, potansiyel olarak yararlı bilgiler sağlayabilir. Önerilen model, rastgele yürüyüşü düğüm tanımlayıcıları yerine bir özellik vektörleri dizisi olarak temsil ederek hem ağın yapısal bilgisinden hem de düğüm özniteliğinden yararlanmayı amaçlamaktadır. Denklem (4.11) ile aşağıdaki gibi ifade edilir.

$$ry = \{\delta(v_1), \delta(v_2), \delta(v_3), \dots, \delta(v_n)\} \tag{4.11}$$

burada $\delta(n)$, düğüm v_{n} için özellik vektörünü temsil eder. Yalnızca bir kategorik özniteliğe sahip bir ağ durumunda, bir düğüm noktası için basit bir özellik vektörü, bir düğüm tanımlayıcısı (id) ve düğüm noktasının özniteliğinden oluşacaktır. Şekil 4.7, her düğüm noktasının kendi özniteliği ile etiketlendiği bir ağı göstermektedir.



Şekil 4.7 Öznitelik bilgisi içeren örnek bir ağ

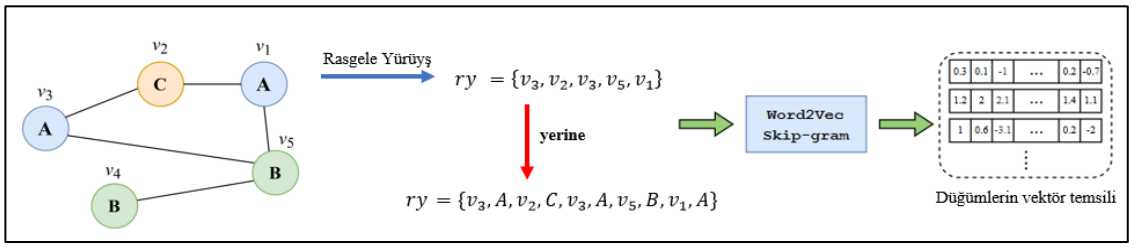
$ry_1 = \{v_3, v_2, v_3, v_5, v_1\}$, v_3 düğümünden başlayan 4 adım uzunluklu örnek bir rasgele yürüyüş olsun. Önerilen yaklaşıma göre, yeni rasgele yürüyüş Denklem (4.12)'deki gibi olacaktır:

$$ry_1 = \{v_3, A, v_2, C, v_3, A, v_5, B, v_1, A\} \tag{4.12}$$

Önerilen modelin (*node2vec-öznitelik*) özellik vektörü Denklem (4.13)'teki bilgilerden oluşacaktır.

$$\delta_{öznitelik}(v) = düğüm\ kimliği\ (id), öznitelik \tag{4.13}$$

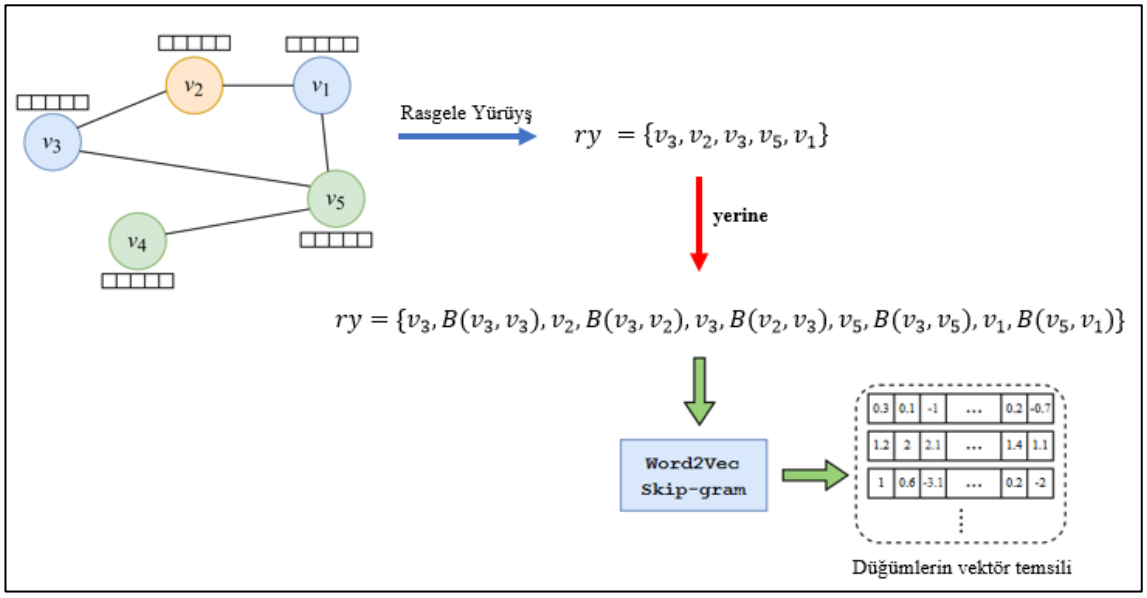
Yürüyüşlere öznitelik bilgilerinin eklenmesinin ardındaki beklenti, genellikle aynı özniteliklerle görünen düğüm noktalarının, farklı öznitelikleriyle oluşan düğümlere göre birbirine daha yakın yerleştirmelere sahip olmasıdır. Düğüm özniteliklerini rasgele yürüyüşlere dahil ederek, Skip-gram yöntemi, eğitim aşamasını özniteliklerle zenginleştirir. Bu nedenle, önerilen yaklaşım yapısal ve öznitelik verilerinin entegrasyonuna izin vermektedir. Şekil 4.8'de, yalnızca bir kategorik özniteliğe sahip bir ağ için önerilen yöntemin (*node2vec-öznitelik*) ardışık düzeni gösterilmektedir.



Şekil 4.8 Önerilen yöntemin (*node2vec-öznitelik*) ardışık düzeni

Önerilen *node2vec-öznitelik* modeli yalnızca bir kategorik özniteliğe sahip ağlar için çalışmaktadır. Örneğin cinsiyet veya yaş gibi sonlu ve ayrık değerlere sahip özniteliklerinden faydalanmaktadır. Ancak gerçek dünyadaki birçok ağ birden fazla değer veya sürekli değerlere sahip öznitelikler içermektedir. Örneğin, sosyal ağdaki bir kullanıcı, profil bilgisi, günlük yaptığı paylaşım sayısı ve paylaşım içeriği gibi birden fazla ve kategorik olmayan bilgilere sahip olabilir. Dolayısıyla oldukça çeşitli olabilen bu

tür zengin öznitelik bilgileri önerilen modele entegre ederek düğüm temsillerinin öğrenimi aşamasında katkı sağlayacaktır. Çok nitelikli olmayan ağlarda rasgele yürüyüşe hangi özniteliğin dahil edileceğini belirtmek gerekli değildir. Çok nitelikli ağlarda ise rasgele yürüyüşler için aynı şey söylenemez. Ayrıca Skip-gram sözcükler yani ayrık girdiler için tasarlanmıştır. Bu nedenle, belirli öznitelikler için belirli model bileşenlerini tasarlayan manuel çabalardan kaçınmak ve rasgele bir yürüyüşe tek bir öznitelik eklemek yerine, en basit yaklaşım, düğümlerin öznitelik benzerliklerini eklemek olacaktır. Bu çalışmada, literatürde çok iyi bilenen 3 farklı benzerlik ölçme yöntemi (Jaccard, Cosine ve Euclidean) kullanılarak önerilen modeldeki rasgele yürüyüşlere tek öznitelik değeri yerine benzerlik değeri eklenmiştir. Birden fazla özniteliğe sahip ağlar için *node2vec-öznitelik+* olarak adlandırılan bu yaklaşımın ardışık düzeni Şekil 4.9'deki gibi olacaktır.



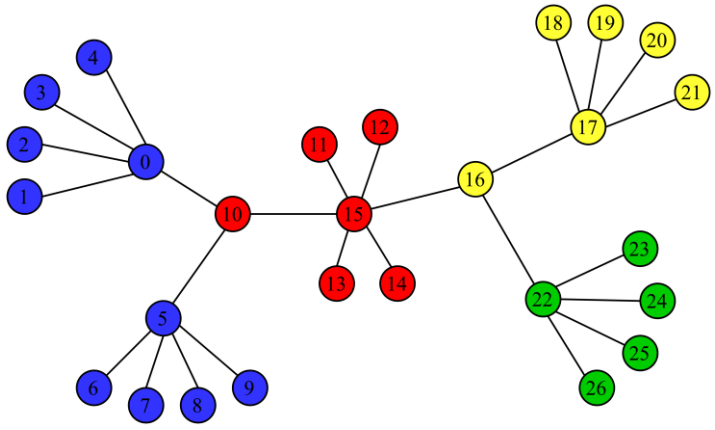
Şekil 4.9 Çoklu özniteliğe sahip ağlar için önerilen yöntemin (*node2vec-öznitelik+*) ardışık düzeni

4.5 Veri Setleri

4.5.1 Topluluk-5

Önerilen modelimizi test etmek amacıyla, 27 düğüm ve 26 bağlantıdan oluşan yapay bir ağ oluşturulmuştur. Topluluk-5 olarak adlandırdığımız bu ağ Şekil 4.10'da

gösterilmektedir. Her düğüm etiketine yeşil, mavi, sarı ve kırmızı rengini atayarak Topluluk-5 ağını oluşturulur. Bu düğüm renkleri, her renk için yerel bir topluluk oluşturulacak şekilde atanır, bu da ağın beş topluluğa bölünmesiyle sonuçlanır, böylece oluşturulan bu yapay ağa Topluluk-5 adı verilmiştir. Ağın düğümleri için renk atamaları aşağıdaki tablo Çizelge 4.1'de gösterilmektedir.



Şekil 4.10 Beş topluluğa sahip Topluluk-5 ağı

Çizelge 4.1 Topluluk-5 ağında düğüm renklerinin atanması

Renk	**Düğümler**
Mavi	0 – 9
Kırmızı	10 – 15
Sarı	16 – 21
Yeşil	22 - 26

4.5.2 Email-EU-core

Email-EU-core (Yin *et al.* 2017), Ekim 2003'ten Mayıs 2005'e kadar (18 ay) olan bir süre boyunca büyük bir Avrupa araştırma kurumundaki üyeler arasındaki e-postadan oluşturulmuş bir ağdır. E-postalar, kurum üyeleri arasında gerçekleşen iletişimi temsil etmektedir. Düğümler kurum üyelerini (benzersiz bir e-posta adresi) ve kenarlar iki üye arasındaki e-postaları temsil eder. Email-EU-core ağı düğüm olarak 1005 e-posta adresi ve kenar olarak 25571 e-posta içermektedir. Veri seti ayrıca her bir düğümün araştırma enstitüsündeki 42 bölümden hangisine ait olduğu bilgisini içermektedir.

4.5.3 Facebook

Doğrulanmış Facebook sayfaları bu ağın temelini oluşturur. Kasım 2017'de Facebook Graph API aracılığıyla toplanmıştır (Rozemberczki *et al.* 2021). Bu ağdaki düğümler, Facebook'taki resmi sayfalardır ve aralarındaki bağlantılar, her iki site arasındaki karşılıklı beğenilerdir. Düğüm öznitelikleri, sayfa sahiplerinin sitenin amacını özetlemek için oluşturdukları site açıklamalarından çıkartılmıştır. Facebook ağı 22470 düğüm ve 171002 bağlantıdan oluşmaktadır. Ayrıca, {politikacılar, hükümet kuruluşları, televizyon kanalları ve şirketler} olmak üzere 4 farklı kategori içermektedir.

4.5.4 Cora

Cora veri seti, düğümlerin bilimsel yayınları, bağlantıların atıfları ve etiketlerin araştırma konularını temsil ettiği atıf ağıdır (Rossi and Ahmed 2015). Her bir bağlantı varlığı, verilen iki yayın *A* ve *B* arasında bir "atıf" ilişkisini gösterir. Cora atıf ağı 2708 düğüm, 5429 link ve 7 farklı sınıftan oluşmaktadır. Her düğümün 1433 boyutlu önceden tanımlanmış bir öznitelik vektörü vardır. Burada 1433, birleştirilmiş tüm yayınlarda bulunan benzersiz kelimelerin sayısıdır. Öznitelik vektöründeki her giriş ya 0 ya da 1 değeri alır; 0, bir yayında o kelimenin yokluğunu ve 1 varlığını temsil eder. Çalışmada kullanılan tüm veri setler Çizelge 4.2'de özetlenmiştir.

Çizelge 4.2 Veri setlerinin istatistiklerine genel bir bakış

	Düğüm sayısı	**Bağlantı Sayısı**	**Etiket Sayısı**	**Gerçekleşen görev**
Topluluk-5	27	26	4	yapısal benzerlik ölçümü
Email-EU-core	1005	25571	42	düğüm sınıflandırma
Facebook	22470	171002	4	düğüm sınıflandırma
Cora	2708	5429	7	düğüm sınıflandırma

4.6 Temel Yöntemler

Önerilen yaklaşımlar, farklı ağ analiz görevleri için tüm veri setlerinde aşağıdaki ilgili temel yöntemlerle karşılaştırılmıştır.

DeepWalk (Perozzi *et al.* 2014), düğüm temsillerini öğrenmek için yapısal bilgiyi kullanan, bilinen bir ağ yerleştirme algoritmasıdır. Düğüm yerleştirme işleminde kesik rasgele yürüyüşleri cümleler olarak, düğümler ise kelimeler olarak kullanır. Rasgele yürüyüşleri analiz etmek için word2vec (Mikolov *et al.* 2013b, a) kelime yerleştirme algoritmasından faydalanır.

Node2vec (Grover and Leskovec 2016), Deepwalk'ın bir uzantısıdır. Bir düğümün çevresine BFS veya DFS keşfini simüle etmek için yürüyüşü kontrol eden iki ekstra parametre (p ve q) kullanarak, Skip-gram'ı taraflı rasgele yürüyüşlerle birleştirir.

LINE (Tang *et al.* 2015), her düğümün birinci dereceden komşularını (lokal) ve ikinci dereceden komşularını (global) örnekleyerek iki tür temsil ayrı ayrı öğrenilir ve nihai düğüm temsili oluşturmak için birleştirilir.

TADW (Yang *et al.* 2015), düğüm temsillerini öğrenmek için hem yapısal bilgiyi hem de düğüm içerik bilgisini kullanır. Her düğümün metin özelliklerini bir matris çarpanlarına ayırma çerçevesi altında yerleştirme işlemine dahil eder.

4.7 Sınıflandırıcı ve Performans Metriği

Bu çalışmada, oluşturulan düğüm yerleştirmelerini düğüm sınıflandırma görevi üzerinde test etmek için lojistik regresyon (logistic regression - LR) sınıflandırıcı kullanılmıştır. Sınıflandırıcıyı oluşturmak için Python'daki scikit-learn makine öğrenimi kütüphanesinden yararlanmıştır.

Lojistik regresyon modeli, bağımlı değişken (bir olayın meydana gelme olasılığı) ile bağımsız değişkenler (olasılığı tahmin etmek için kullanılan değişkenler) arasındaki ilişkiyi tanımlayan (tahmin eden) en uygun eğriyi bulma fikrine dayanır. İki veya ikiden fazla kategoriye sahip kategorik bir sonucun olasılığını tahmin etmek için kullanılan istatistiksel bir tekniktir. Öngörülen olasılığı 0 ile 1 arasında bir değere eşleyen bir sigmoid eğrisi olan lojistik fonksiyona dayanır. Genellikle müşteri davranışını tahmin

etmek, tıbbi bir durum olasılığını tahmin etmek ve kredi temerrüdü olasılığını tahmin etmek dahil olmak üzere çok çeşitli uygulamalarda kullanılır (Sammut and Webb 2011). Lojistik regresyon için genel formül Denklem (4.14) ve Denklem (4.15)'teki gibi tanımlanır:

$$z = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n \tag{4.14}$$

$$p = \frac{1}{1 + e^{(-z)}} \tag{4.15}$$

Burada p meydana gelen olayın tahmin edilen olasılığı temsil eder ve b_0 kesme terimidir. $(x_1, x_2, \ldots x_n)$ bağımsız değişkenlerdir ve $(b_1, b_2, \ldots b_n)$ bağımsız değişkenler için katsayılardır. Bu formülde, bağımsız değişkenler $(x_1, x_2, \ldots x_n)$, karşılık gelen katsayıları $(b_1, b_2, \ldots b_n)$ ile çarpılır ve kesme terimi (b_0) ile toplanır. Sonuç, lojistik fonksiyondan $(1/1 + e^{(-z)})$ geçirilir, burada z, bağımsız değişkenlerin ve kesme teriminin toplamının sonucudur. Lojistik fonksiyonun çıktısı, 0 ile 1 arasında değişen, meydana gelen olayın tahmin edilen olasılığıdır.

Önerilen ve temel yöntemlerin performansını değerlendirmek için $F1 - score$ metriği kullanılmıştır. $F1 - score$, Doğruluk (Accuracy) metriğinde yapılan genel bir performanstan ziyade sınıf bazında performans değerlendirmesi gerçekleştirir. Hem Precision'e (Hassasiyet) hem de Recall'a (Duyarlık) eşit ağırlık verir, bu da onu diğer metriklere göre çok iyi bir alternatif yapmıştır (Yacouby and Axman 2020). $F1 - score$ metriği Denklem (4.16)'daki gibi tanımlanır:

$$F1 - score = \frac{2TP}{2TP + FP + FN} \tag{4.16}$$

burada TP doğru şekilde "pozitif" olarak tahmin edilen numune sayısını, FP yanlış bir şekilde "pozitif" olarak tahmin edilen numune sayısını ve FN yanlış bir şekilde "negatif" olarak tahmin edilen numune sayısını temsil eder.

5. BULGULAR VE TARTIŞMA

Bu bölümde önerilen yaklaşımların yapısal rollerin benzerliği ve düğüm sınıflandırması üzerindeki etkinliğini değerlendirmek için gerçek dünya veri setleri üzerinde testler gerçekleştirilmiştir. Bu analitik görevler, oluşturulan yerleştirmelerin düğüm öznitelik bilgisinden faydalanıp faydalanmadığını anlamak için içgörü sağlamaktadır.

5.1 Yapısal Rollerin Benzerliği

Ağlarda rol belirleme fikri, yapısal benzerlik çatısı altında yer alan kavramlardan biridir. Gerçek dünya ağlarında sıklıkla meydana gelen belirli yapısal kalıplara roller denir (Henderson *et al.* 2012). Gerçek dünya ağlarında karşılaşacağımız Hub (merkez düğüm), Periphery (çevre düğüm) ve Middle (ara düğüm) olmak üzere 3 farklı rol bulunmaktadır (Joyce *et al.* 2010). Hub (merkez düğüm), ağda son derece bağlantılı ve komşularından çok daha yüksek bir dereceye sahip olan bir düğüm noktasıdır. Ağın kenarlarında bulunan çevresel düğümler (Periphery) genellikle daha düşük derecelidir ve yalnızca merkezi ve ara düğümlere (Middle) bağlıdır. Ara düğüm kategorisine giren düğüm noktaları, ne merkezi ne de çevre olanlardır. Bu bölümde, önerilen *node2vec-öznitelik* modelimizi kullanarak node2vec'e öznitelik eklemenin etkisi araştırılmıştır. Spesifik olarak amaç, hem yapısal hem de anlamsal benzerliği etkili bir şekilde entegre edip etmediğini belirlemektir. Ayrıca, *node2vec-öznitelik* modelin kümeleme ve çok boyutlu ölçekleme (MDS) sonuçlarını inceleyerek node2vec'in sonuçlarıyla karşılaştırmalı bir analiz yapmayı amaçlamaktadır. Bunun için yapısal rollerin benzerlik testi, bölüm (4.5.1)'de daha ayrıntılı olarak açıklanan Topluluk-5 ağı üzerinde gerçekleştirilmiştir. Çizelge 5.1 Topluluk-5 ağındaki düğümlerin yapısal rolleri manuel olarak üç yapısal role göre nasıl sınıflandırıldığını göstermektedir.

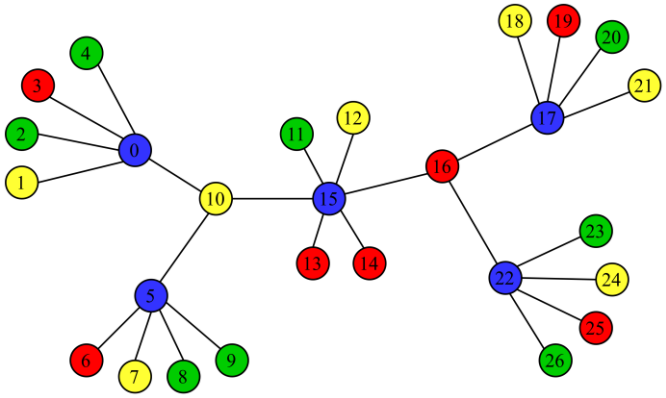
Çizelge 5.1 Topluluk-5 ağındaki düğümlerin yapısal rolleri

Yapısal roller	Düğümler
Hubs (merkez düğümler)	0, 5, 15, 17, 22
Periphery (çevre düğümler)	1, 2, 3, 4, 6, 7, 8, 9, 11, 12, 13, 14, 18, 19, 20, 21, 23, 24, 25, 26
Middle (ara düğümler)	10, 16

Node2vec'in yapısal benzerliği tespit edebildiğini doğrulamak için orijinal makalede yazarların önerdiği aynı p ve q parametreleri kullanılmıştır. Diğer parametreler belirtilmediği için deneme yanılma yoluyla en iyi temsilleri oluşturan parametreler seçilmiştir. Node2vec ve *node2vec-öznitelik* modellerin Topluluk-5 ağında yapısal rol benzerliklerini bulabileceği parametreler kümesi aşağıda verilmiştir:

- $p = 1$ ve $q = 2$
- $nw = 10$ (Rasgele yürüyüş sayısı)
- $wl = 5$ (Rasgele yürüyüş uzunluğu)
- $ws = 3$ (Pencere boyutu)
- $d = 8$ (Temsil boyutu)

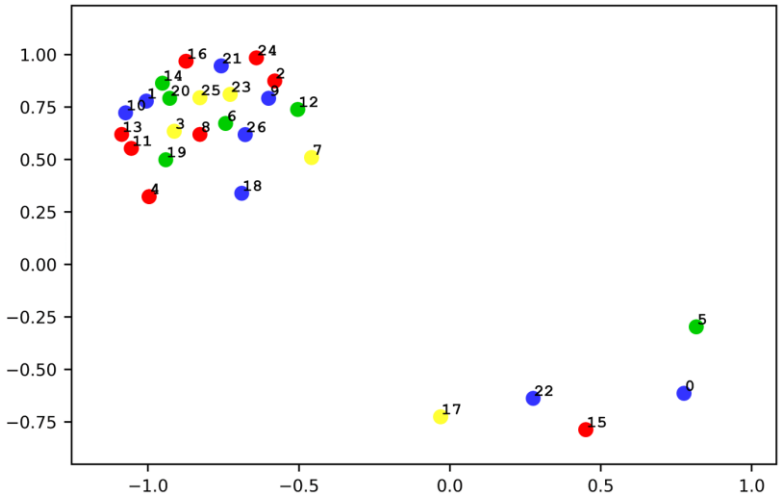
Node2vec ve *node2vec-öznitelik* modelleri üç yapısal rolü de ayırt edip edemediğini test etmek için, önce düğümlerin vektör temsilleri oluşturulmuş sonra k-ortalama (k-means) algoritmasını kullanarak kümelemelerin Şekil 4.10'daki topluluklara karşılık gelip gelmediği araştırılmıştır. En son olarak modellerin ağdaki üç yapısal rol ayrımını değerlendirmek için, çok boyutlu ölçekleme (MDS) düğüm temsillerinin 2 boyutlu görselleri çizdirilmiştir.



Şekil 5.1 Node2vec ile k-ortalama (k=4) kümeleme sonucu

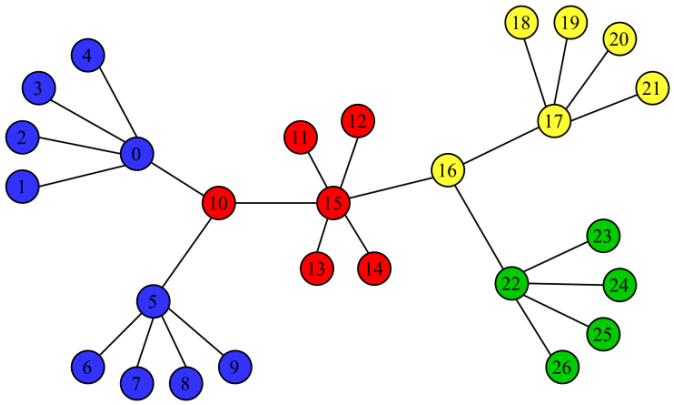
Karşılaştırma için önce node2vec test edilmiştir. 4 farklı renkten {yeşil, mavi, sarı, kırmızı} oluşan Topluluk-5 ağının düğüm etiketlerini doğru bir şekilde kümelemek için k = 4 seçilmiştir. Node2vec'in k = 4 ile k-ortalama kümeleme çıktısı Şekil 5.1'de gösterilmektedir. Renkler kümeleri temsil etmektedir. Şekil 5.2 ise, gerçek topluluklarına

göre renklendirilmiş düğüm noktalarla ilgili MDS görselini göstermektedir. Sonuç olarak, Şekil 5.1'de diğer yapısal rollerin düğümleri karışırken merkez düğümler olan {0, 5, 15, 17 ve 22} bir arada kümelendiği görülebilir. Şekil 5.2'de ise, ara düğümler (10 ve 16) çevre düğümleriyle birlikte karıştırılırken, merkez düğümlerin diğer düğümlerden net bir şekilde ayrıldığı MDS görselinde de görülebilir. Başka bir deyişle, node2vec ile kısmi yapısal benzerlik tespit edilmiştir.

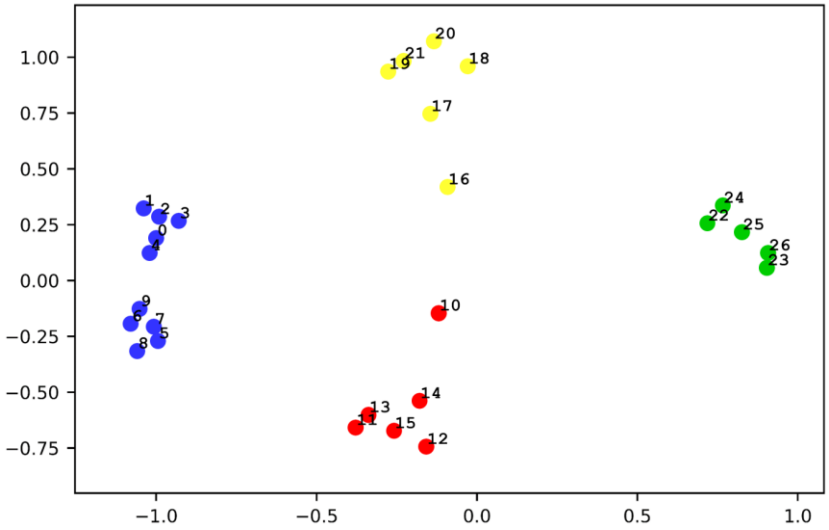


Şekil 5.2 Node2vec ile elde edilen düğüm temsillerin iki boyutlu gösterimi

Şekil 5.3, k = 4 için k-ortalama ile kümeleme yapıldığında *node2vec-öznitelik* sonucunu ve Şekil 5.4'teki karşılık gelen MDS görselini göstermektedir. Bu testler için, özniteliklerin dahil edilmesini hesaba katmak için pencere boyutu iki katına çıkarılarak pencere boyutu 6 seçilmiştir.



Şekil 5.3 *Node2vec-öznitelik* ile k-means (k=4) kümeleme sonucu



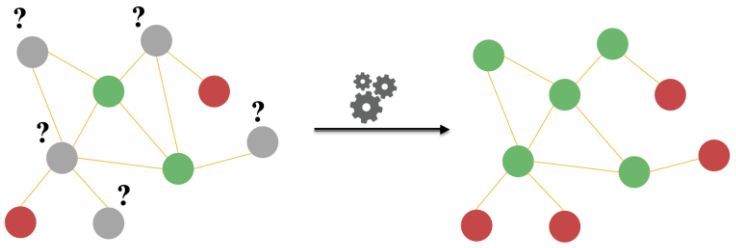
Şekil 5.4 *Node2vec-öznitelik* ile elde edilen düğüm temsillerin iki boyutlu gösterimi

Şekil 5.3'te ortaya çıkan kümeleme, orijinal Topluluk-5 ağının topluluklarıyla aynıdır. İki mavi topluluğun düğümleri, anlamsal benzerliği düşündürecek şekilde birlikte kümelenmiştir. Bununla birlikte, Şekil 5.4'te ise iki mavi topluluk arasında hafif bir ayrım vardır, bu da sonuçların bir şekilde yapısal yakınlığa dayalı olabileceği anlamına gelir. Ayrıca, (10 ve 16) ara düğümler dışında, tüm düğüm noktaları renk topluluklarına nispeten yakındır. Ortadaki (10 ve 16) düğümleri arasında biraz mesafe vardır, ancak yine de ait oldukları renk topluluklarına nispeten daha yakındırlar. Sonuç olarak, *node2vec-öznitelik* yöntemindeki kümeleme hem yapısal hem de anlamsal benzerliği hesaba katarak orijinal node2vec yönteminden daha iyi kümeleme yapmaktadır.

5.2 Düğüm Sınıflandırması

Düğüm sınıflandırması klasik bir ağ tabanlı problemdir. Düğüm sınıflandırması, düğüm temsillerinin kalitesini doğrulamak için popüler bir yöntem haline gelmiştir (Goyal and Ferrara 2018). Düğümlere ait olduğu bir dizi etiket verildiğinde, amaç, bazı bilgi çıkarma ve analiz tabanlı algoritmaya dayalı olarak düğümleri ilgili kategorilerine göre sınıflandırmaktır (Bhagat *et al.* 2011). Örneğin, her biri beş kategoriden birine ait olan bir dizi araştırma makalesi verildiğinde, amaç her makaleyi ait olduğu ilgili kategoriye göre sınıflandırmaktır. Sınıflandırma, her makalenin metnine, konusuna veya yazarlarına bağlı

olabilir. Şekil 5.5, makine öğrenimi kullanılarak tahmin edilebilecek eksik etiketlere sahip bir ağı göstermektedir.



Şekil 5.5 Eksik etiketler için düğüm sınıflandırması

Bir düğüm sınıflandırma görevi aşağıdaki gibi özetlenebilir. $D = \{(v_1, c_1), \dots, (v_n, c_n)\}$ etiketli düğümlerden oluşan bir eğitim seti verildiğinde ve $c_i \in C = \{c_1, c_2, \dots, c_m\}$ bir sınıf etiketi olsun. Amaç, sınıfları etiketlenmemiş düğümleri uygun sınıflara atayabilen bir sınıflandırma modeli geliştirmektir. Etiketlenmemiş düğümleri sınıflara eşleyen bir sınıflandırma modeli eğitmek olacaktır. Bu çalışmada, eğitim verilerinin %70'ten başlayarak %10'luk bir artışla, her yöntem için üç farklı (eğitim - test) kümesi gerçekleştirilmiştir. Sınıflandırma modeli için doğrudan ana sınıflandırıcı olarak lojistik regresyonu ve performans ölçümü olarak $F1 - score$ ölçütü kullanılmıştır. Node2vec tabanlı yöntemler için, tüm olası $p, q \in \{0{,}25\ , 0{,}5\ , 1\ , 2\ , 4\}$ kombinasyonları için (80 - 20)% eğitim-test ayrımında yöntemleri eğiterek optimal p ve q değerleri belirlenmiştir. Optimum p ve q değerleri Çizelge 5.2'de verilmiştir. Diğer parametreler ise:

- $nw = 25$ (Rasgele yürüyüş sayısı)
- $wl = 15$ (Rasgele yürüyüş uzunluğu)
- $ws = 5$ (Pencere boyutu)
- $d = 64$ (Temsil boyutu)

Çizelge 5.2 Node2vec tabanlı yöntemler için seçilen $\boldsymbol{p}$ ve $\boldsymbol{q}$ değerleri

Yöntemler	**(p, q)**
node2vec	(2, 1)
node2vec-öznitelik	(2, 0,5)
node2vec-öznitelik+	(1, 0,5)

Diğer yöntemler için varsayılan parametreler seçilmiştir. Email-EU-core ve Facebook ağlarının düğümleri tek özniteliğe sahip oldukları için önerilen *node2vec-öznitelik* yöntemimizi diğer temel yöntemlere karşı düğüm sınıflandırma performansı değerlendirilmiştir. Email-EU-core ve Facebook ağlarındaki bölüm ve sınıf etiketleri sırasıyla öznitelik olarak kullanılmıştır. Çizelge 5.3, önerdiğimiz *node2vec-öznitelik* modelin tutarlı bir şekilde diğer temel yöntemlerden daha iyi performans gösterdiğini göstermektedir; bu da düğüm özniteliğin daha iyi düğüm temsillerini oluşturmak için ne kadar faydalı olduğunu doğrulamaktadır. Bu deneysel çalışmada TADW yöntemin performansı test edilmemiştir. Çünkü TADW yöntemi düğüm temsillerini oluşturma aşamasında hem yapısal bilgiyi hem de düğüm içerik bilgisinden yaralanmasına rağmen her iki veri setinin düğümleri metin öznitelikleri içermemektedir.

Çizelge 5.3 Tek özniteliğe sahip ağların düğüm sınıflandırma performansı

Veri seti	**Yöntem**	**(70 – 30)%**	**(80 -20)%**	**(90 – 10)%**
Email-EU-core	DeepWalk	78,1	78,1	78,4
	node2vec	77,6	77,8	77,8
	LINE	76,8	77,6	77,4
	node2vec-öznitelik	87,8	88,2	88,0
Facebook	DeepWalk	82,3	82,5	82,4
	node2vec	83,0	83,4	83,1
	LINE	79,7	80,0	80,2
	node2vec-öznitelik	85,1	85,6	85,5

İkinci düğüm sınıflandırma testi Cora veri seti üzerinde gerçekleştirilmiştir. Cora ağının her bir düğümü 1433 boyutlu önceden tanımlanmış bir öznitelik vektörü içermektedir. Bu nedenle *node2vec-öznitelik* yerine *node2vec-öznitelik+* yöntemimizi diğer temel yöntemlere karşı düğüm sınıflandırma görevi üzerinde performansı değerlendirilmiştir. Ayrık olmayan bu öznitelik bilgileri ilk önce 3 farklı benzerlik ölçme yöntemi (Jaccard, Cosine ve Euclidean) kullanılarak *node2vec-öznitelik+* modelindeki rasgele yürüyüşlere tek öznitelik değeri yerine benzerlik değeri eklenmiştir. Bir önceki deneysel çalışmada her yöntem için genel olarak en yüksek başarıyı (80 - 20)% (eğitim-test) ayrımında alındığından dolayı bu deneysel çalışma sadece (80 - 20)% (eğitim-test) kümesi üzerinde gerçekleştirilmiştir.

Cora ağına ait sonuçlar Çizelge 5.4'te özetlenmiştir. Önerilen *node2vec-öznitelik+* modelimiz farklı benzerlik ölçümlerinde diğer yöntemlerden daha iyi performans göstermektedir. TADW, yapısal bağlam ve düğüm içeriği arasındaki etkileşimleri yakaladığı için en iyi ikinci performansı göstermektedir. Bununla birlikte, TADW sadece düğüm metin özelliklerini dikkate aldığı ve bu özellikleri matris çarpanlarına ayırma çerçevesi altında yerleştirme işlemine dahil ettiği için, *node2vec-öznitelik+* tarafından öğrenilen düğüm temsilleri kadar etkili olmayabilir. Bu, özellikle ağ temsili öğrenimi için düğüm içeriğinden yararlanıldığında, sadece tek bir öznitelik türünden değil çeşitli öznitelik türlerinin korunması gereken önemli bir özellik olduğunu göstermektedir. Ayrıca TADW matris üzerinde çarpanlara ayırma işlemi gerektirdiği için alan ve hesaplama açısından pahalıdır ve hatta bazen mümkün değildir. *Node2vec-öznitelik+* modeli Jaccard benzerliğiyle birlikte kullanıldığında diğer benzerlik ölçümlerine göre daha iyi performans göstermektedir. Bu başarı Jaccard benzerliğinin temelinde yatan her iki vektörlerinin kesişimdeki toplam elemanlar ($|\boldsymbol{x} \cap \boldsymbol{y}|$) temsil etmesiyle ilişkilendirilebilir.

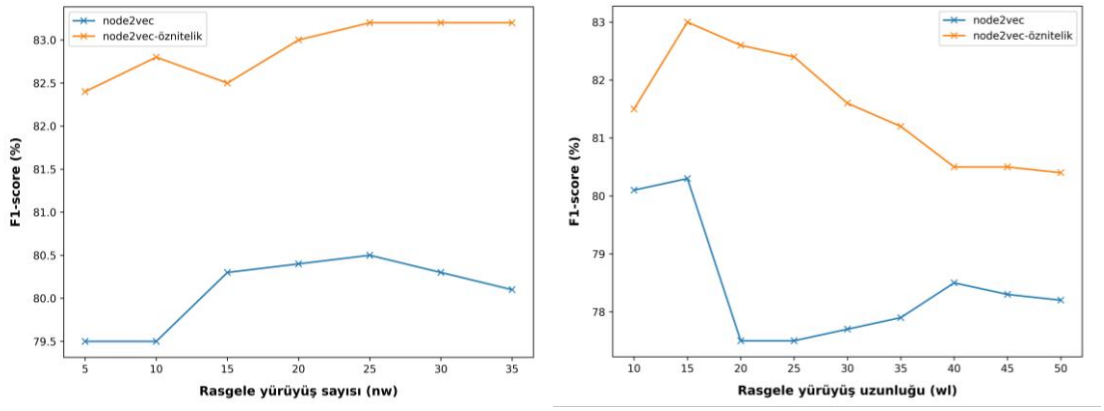
Çizelge 5.4 Cora ağı için farklı yöntemlerin düğüm sınıflandırma performansı

Veri seti	**Yöntem**		**Doğruluk Oranı (%)**
Cora	DeepWalk		79,4
	node2vec		80,6
	LINE		77,5
	TADW		81,3
	node2vec-öznitelik+	Cosine	83,2
		Jaccard	<u>83,5</u>
		Euclidean	82,8

Genel olarak sunulan sonuçlar, önerilen yaklaşımlar ağ ile ilişkili yapısal ve anlamsal bilgileri yakalama ve kodlama konusundaki etkinliğini ve sağlamlığını ortaya koymaktadır.

5.3 Parametre Hassasiyeti

Node2vec orijinal makalesinde (Grover and Leskovec 2016), rasgele yürüyüş uzunluğu wl veya rasgele yürüyüş sayısı nw değiştiğinde düğüm temsillerin kalitesine etkisi olmuştur. Bu nedenle node2vec ve *node2vec-öznitelik* yöntemlerin iki parametresinin, rasgele yürüyüş uzunluğu ve rasgele yürüyüş sayısının Email-EU-core ağı üzerindeki etkisi araştırılmıştır. Bu deneylerde ($p = q = 1$) ayarlanmıştır. İlk deneyde rasgele yürüyüş sayısı nw parametresi (5, 35) aralığında seçilirken diğer rasgele yürüyüş uzunluğu wl parametresi 15 olarak sabitlenmiştir. İkinci deneyde ise, rasgele yürüyüş uzunluğu wl parametresi (10, 50) aralığında seçilirken diğer rasgele yürüyüş sayısı nw parametresi 25 olarak sabitlenmiştir. Sınıflandırıcı olarak lojistik regresyon kullanılmıştır. Deneylerin düğüm sınıflandırma sonucu Şekil 5.6'de gösterilmektedir.



Şekil 5.6 Node2vec tabanlı yöntemlerin yürüyüş uzunluğu ve yürüyüş sayısı parametlerinin değişimindeki performansı

Rasgele yürüyüş uzunluğunu arttırıldığında node2vec ve *node2vec-öznitelik* performansının düşme eğiliminde olduğu gözlenmiştir. Rasgele yürüyüş uzunluğu 15'ten 20'ye çıktığında, *node2vec-öznitelik* performansta büyük düşüşe sahiptir. Öte yandan rasgele yürüyüş sayısı artırıldığında her iki yöntemin performansı istikrarlı bir şekilde yükselmiştir. Node2vec ve *node2vec-öznitelik* yöntemleri, düşük yürüyüş sayılarında (20 - 25) ve düşük rasgele yürüyüş uzunluklarında (15) en iyi performansı elde etmektedir.

5.4 Tartışma ve Değerlendirme

Yerleştirme uzayında orijinal ağın farklı gizli özelliklerini yakalamak zor olduğundan, ağ temsili öğrenmeyi zorlu bir problem haline getirmektedir. Çıktı yerleştirmenin içsel özellikleri yeterince yakalayamaması durumunda, çeşitli tahmin görevlerindeki performansı yetersiz kalabilir. Tek bir bilgi, tüm tahmin görevleri için düğüm yerleştirmesini genelleştirmeyebilir. Örneğin, struc2vec yapısal benzerlik tahmini için iyi performans gösterirken (Ribeiro *et al.* 2017), DeepWalk homofili tahmini için iyi performans göstermektedir (Perozzi *et al.* 2014). Bazı yöntemler birden fazla ağ bilgisinden faydalanarak yüksek kaliteli bir düğüm yerleştirmesi oluşturabilir ancak karmaşık ve daha yüksek hesaplama gücüne sahip olacaktır. Çizelge 5.5'te, Cora veri seti üzerinde önerilen *node2vec-öznitelik* modelimiz ile altı farklı çalışmanın (düğüm yerleştirme yönteminin) çeşitli özelliklere göre düğüm sınıflandırma doğruluğu özetlenmektedir. Diğer yöntemlerin doğruluk değerleri orijinal çalışmalardan alınmıştır.

Çizelge 5.5 Cora ağı için düğüm sınıflandırma görevinde model başarı sonuçları

Çalışmalar	Kategori	Topoloji	Öznitelik	Etiket	Doğruluk
GraRep (Cao *et al.* 2015)	Matris çarpanlara ayırma	✓			80,89
DMF (Zhang *et al.* 2016a)		✓	✓	✓	82,91
GAT (Veličković *et al.* 2017)	Derin öğrenme	✓	✓	✓	83,0 ± 0,7
GCN-LPA (Wang and Leskovec 2020)		✓	✓	✓	88,5 ± 1,5
GW (Fazaeli and Momtazi 2022)	Olasılık (Rasgele yürüyüş)	✓		✓	84,0
GERI (Sun and Zhang 2017)		✓	✓	✓	87,0
node2vec-öznitelik+		✓	✓		88,4

Çizelge 5.5, matris çarpanlara ayırma dayalı yöntemlerinin diğer yaklaşımlarla karşılaştırıldığında daha düşük doğruluk skorlarına sahip olduğunu göstermektedir. Ayrıca, bu yöntemler komşuluk/benzerlik matrisi üzerinde çarpanlara ayırma işlemi gerektirdiği için alan ve hesaplama açısından pahalıdır ve ölçekli ağ verileri işlemekle

sınırlıdır (Grover and Leskovec 2016). Ayrımcı matris çarpanlara ayırma (discriminative matrix factorization - DMF) yaklaşımı (Zhang *et al.* 2016a) ağ içerisindeki topoloji, öznitelik ve etiket bilgilerini düğüm yerleştirmesine entegre etmesine rağmen diğer kategorideki yöntemlerden daha düşük performans göstermektedir. Bunun nedeni, bu yöntemin optimizasyon aşamasında kullandığı sabit parametrelerinin Cora veri seti için uygun olmamasından kaynaklanabilir. Derin öğrenme tabanlı yöntemler, derin ve karmaşık sinir ağları kullandığı için yüksek düzeyde doğrusal olmayan veri yapısını çok iyi yakalar. GAT modeli matris çarpanlara ayırma dayalı yöntemlere kıyasla daha iyi performans göstermektedir. Bunun nedeni, modelin dikkat mekanizması (attention mechanism) aracılığıyla farklı komşu düğümlerin önemini tam olarak dikkate almasıdır (Veličković *et al.* 2017). Çizelge 5.5'te listelenen sonuçlardan, etiket yayılımı ve çizge konvolüsyonel sinir ağları (label propagation and graph convolutional neural networks – GCN-LPA) modelinin en iyi performans gösteren yaklaşım olduğu gözlenmektedir. Bunun nedeni, bu yöntemin optimizasyon aşamasında yapısal ve düğüm öznitelik bilgilerini düğüm etiket bilgisiyle daha iyi birleştirebilmesidir (Wang and Leskovec 2020). Ancak düğüm öznitelik ve etiket bilgilerine duyulan ihtiyaç, derin öğrenme yaklaşımların düğüm öznitelik ve etiket bilgileri olmayan homojen ağlara uygulanması zordur (Waikhom and Patgiri 2021). Genel olarak sonuçlar, rasgele yürüyüş tabanlı yöntemlerin (rehberli yürüyüş (guided walk - GW), zengin bilgiler içeren çizge yerleştirme (graph embedding with rich information - GERI) ve *node2vec-öznitelik+*), ağlardaki bağlamsal bilgileri yakalayabildikleri için üstün performans göstermektedir. Ayrıca bu başarı, rasgele yürüyüş ile düğümlerin lokal ve global bilgilerini yakalayabilen BFS ve DFS stratejilerinin avantajlarına bağlanabilir.

6. SONUÇ VE ÖNERİLER

Özellik öğrenme teknikleri, ağdaki yapısal bilgiyi koruyarak her düğüm için düşük boyutlu bir vektör temsilini öğrenmeyi amaçlar. Gerçek dünya ağları genellikle çok sayıda düğüme ve yüksek boyutlu düğüm özniteliklerine sahiptir. Düğüm öznitelikleri, günümüzün gerçek dünya ağlarında bulunan kritik bilgilerdir. Mevcut özellik öğrenme teknikleri, düğüm temsillerini öğrenmek için yalnızca ağ yapısından faydalanmaktadır.

Bu çalışmada, ağ temsillerini öğrenmek için ağın yapısal ve düğüm içeriğinden gelen bilgileri etkili bir şekilde kullanmasına izin veren birden çok node2vec uzantısı önerilmiştir. Önerilen yaklaşımlar, bir ağı etkili bir şekilde iki bilgi kaynağı arasındaki etkileşimi yakalayan tek bir düşük boyutlu temsil alanına yerleştirmektedir. Öğrenilen temsiller hem ağ yapısını hem de düğüm içerik bilgilerini korumaktadır. Önerilen ilk *node2vec-öznitelik* modeli yalnızca bir kategorik özniteliğe sahip ağlar için uygulanmaktadır. İkinci *node2vec-öznitelik+* modeli ise, 3 farklı benzerlik ölçme yönteminden (Jaccard, Cosine ve Euclidean) faydalanarak önerilen modeldeki rasgele yürüyüşlere tek öznitelik değeri yerine çok nitelikli düğümlerin benzerlik değerini eklemektedir.

Önerilen yaklaşımların yapısal rollerin benzerliği ve düğüm sınıflandırması üzerindeki etkinliği gerçek dünya ağları üzerinde test edilmiştir. Gerçekleştirilen tüm deneysel çalışmalar, her bir algoritmanın etkinliğini kanıtlamaktadır. Önerilen modeller hem tek kategorik özniteliğe sahip hem de çok nitelikli ağlar için etkinliğini ve verimliliğini göstermektedir. Genel olarak sunulan sonuçlar, önerilen modellerin ağ ile ilişkili düğüm içerik bilgilerini yakalama ve kodlama konusundaki etkinliğini ve sağlamlığını ortaya koymaktadır.

Gerçek dünyada ağların yaygınlığı göz önüne alındığında, ağ verisi üzerinde temsil öğrenimi, düğüm sınıflandırma, bağlantı tahmini ve anormallik tespiti gibi çok çeşitli uygulamalarda önemli bir rol oynamaktadır. Bununla birlikte, modern makine öğrenimi algoritmaları basit diziler (örneğin, sabit boyutlu görüntüler veya metin) için tasarlanmıştır ve ağlar genellikle karmaşık topografik yapılara ve çok modelli özelliklere

sahiptir. Düğüm yerleştirme algoritmaları ile düğümlerin gizli özellik temsilleri öğrenilerek ağ verilerinin vektörleştirilmesi yapılabilir. Ayrıca bu yöntemler, ağın altında yatan topoloji ve düğüm öznitelik hakkında yeterli bilgileri yakalayabildiği için gelecekteki eksik bağlantıları tahmin edebilmektedir. Örneğin, sosyal ağlar reklam amaçlı olduğu kadar arkadaşlık veya içerik önerileri gibi uygulamalar için de kullanılmaktadır. Son olarak, önerilen yöntemin büyük ağ veri üzerindeki performansını test etmek ve verimliliği önemli ölçüde artırabilecek bir yol sunulabilir. Ayrıca, önerilen yöntemin ağırlıklı ve yönlü ağlarda kullanılabilme seçeneğine sahip olması sağlanabilir.

KAYNAKLAR

Ahmed, N. K., Rossi, R. A., Lee, J. B., Willke, T. L., Zhou, R., Kong, X. and Eldardiry, H. 2020. Role-based graph embeddings. IEEE Transactions on Knowledge and Data Engineering, 34(5): 2401-2415.

Barrat, A., Barthelemy, M. and Vespignani, A. 2008. Dynamical processes on complex networks, Cambridge University Press, 361 pages, UK.

Belkin, M. and Niyogi, P. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering, In: Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, pp. 585-591.

Bengio, Y., Ducharme, R. and Vincent, P. 2000. A neural probabilistic language model. The Journal of Machine Learning Research, 3: 1137-1155.

Bhagat, S., Cormode, G. and Muthukrishnan, S. 2011. Node classification in social networks. Social Network Data Analytics, 5: 115-148.

Boffey, T. 1982. Graph theory in operations research, Springer, 301 pages, London.

Breitkreutz, B., Stark, C. and Tyers, M. 2003. Osprey: A network visualization system. Genome Biology, 4: 1-4.

Brin, S. 1998. The pagerank citation ranking: Bringing order to the web. Proceedings of ASIS, 98: 161-172.

Bronstein, M. M., Bruna, J., Lecun, Y., Szlam, A. and Vandergheynst, P. 2017. Geometric deep learning: Going beyond euclidean data. IEEE Signal Processing Magazine, 34(4): 18-42.

Cao, S., Lu, W. and Xu, Q. 2015. Grarep: Learning graph representations with global structural information, In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 891-900.

Cao, S., Lu, W. and Xu, Q. 2016. Deep neural networks for learning graph representations, In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, pp. 1145-1152.

Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C. and Murphy, K. 2022. Machine learning on graphs: A model and comprehensive taxonomy. Journal of Machine Learning Research, 23(89): 1-64.

Chen, J., Zhang, Q. and Huang, X. 2016. Incorporate group information to enhance network embedding, In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 1901-1904.

Choi, S., Cha, S. and Tappert, C. C. 2010. A survey of binary similarity and distance measures. Journal of Systemics, Cybernetics and Informatics, 8(1): 43-48.

Clauset, A., Newman, M. E. and Moore, C. 2004. Finding community structure in very large networks. Physical Review E, 70(6): 066111.

Collobert, R. and Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning, In: Proceedings of the 25th International Conference on Machine Learning, pp. 160-167.

Erdős, P. and Rényi, A. 1960. On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci., 5(1): 17-60.

Euler, L. 1956. The seven bridges of königsberg. The World of Mathematics, 1 573-580.

Fazaeli, M. and Momtazi, S. 2022. Guidedwalk: Graph embedding with semi-supervised random walk. World Wide Web, 25(6): 2323-2345.

Gallagher, B. and Eliassi-Rad, T. 2010. Leveraging label-independent features for classification in sparsely labeled networks: An empirical study, In: Advances in Social Network Mining and Analysis: Second International Workshop, pp. 1-19, Las Vegas.

Gao, L., Yang, H., Zhou, C., Wu, J., Pan, S. and Hu, Y. 2018. Active discriminative network representation learning, In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 2142-2148.

Gomez-Rodriguez, M., Leskovec, J. and Krause, A. 2012. Inferring networks of diffusion and influence. ACM Transactions on Knowledge Discovery from Data (TKDD), 5(4): 1-37.

Goyal, P. and Ferrara, E. 2018. Graph embedding techniques, applications, and performance: A survey. Knowledge-Based Systems, 151: 78-94.

Grohe, M. 2020. Word2vec, node2vec, graph2vec, x2vec: Towards a theory of vector embeddings of structured data, In: Proceedings of the 39th ACM Symposium on Principles of Database Systems, pp. 1-16.

Grover, A. and Leskovec, J. 2016. Node2vec: Scalable feature learning for networks, In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855-864.

Hamilton, W. L., Ying, R. and Leskovec, J. 2017. Representation learning on graphs: Methods and applications. arXiv preprint arXiv:1709.05584.

Henderson, K., Gallagher, B., Eliassi-Rad, T., Tong, H., Basu, S., Akoglu, L., Koutra, D., Faloutsos, C. and Li, L. 2012. Rolx: Structural role extraction and mining in large graphs, In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1231-1239.

Huisman, M. and Van Duijn, M. 2004. Software for statistical analysis of social networks, In: The Sixth International Conference on Logic and Methodology, Amsterdam.

Joyce, K. E., Laurienti, P. J., Burdette, J. H. and Hayasaka, S. 2010. A new measure of centrality for brain networks. PLoS One, 5(8): 12200.

Khanam, K. Z., Singhal, A. and Mago, V. 2021. Noddle: Node2vec based deep learning model for link prediction, In: National Conference on Big Data Technology and Applications, pp. 196-212.

Knoke, D. and Yang, S. 2019. Social network analysis, SAGE Publications, 200 pages.

Kumar, R., Novak, J. and Tomkins, A. 2006. Structure and evolution of online social networks, In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 611-617.

Kumar, S. and Hanot, R. 2021. Community detection algorithms in complex networks: A survey, In: Advances in Signal Processing and Intelligent Recognition Systems: 6th International Symposium, pp. 202-215, Chennai, India.

Li, X., Du, N., Li, H., Li, K., Gao, J. and Zhang, A. 2014. A deep learning approach to link prediction in dynamic networks, In: Proceedings of the 2014 SIAM International Conference on Data Mining, pp. 289-297.

Likaj, R., Shala, A., Mehmetaj, M., Hyseni, P. and Bajrami, X. 2013. Application of graph theory to find optimal paths for the transportation problem. IFAC Proceedings Volumes, 46(8): 235-240.

Lü, L. and Zhou, T. 2011. Link prediction in complex networks: A survey. Physica A: Statistical Mechanics and its Applications, 390(6): 1150-1170.

Ma, X. and Hovy, E. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. arXiv preprint arXiv:1603.01354.

Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N. and Czajkowski, G. 2010. Pregel: A system for large-scale graph processing, In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data, pp. 135-146.

Malliaros, F. D. and Vazirgiannis, M. 2013. Clustering and community detection in directed networks: A survey. Physics Reports, 533(4): 95-142.

Martella, C., Shaposhnik, R., Logothetis, D. and Harenberg, S. 2015. Practical graph analytics with apache giraph, Apress, 315 pages, Berkeley.

Mason, O. and Verwoerd, M. 2007. Graph theory and networks in biology. IET. Syst. Biol., 1(2): 89-119.

Menon, A. K. and Elkan, C. 2011. Link prediction via matrix factorization, In: Machine Learning and Knowledge Discovery in Databases: European Conference, pp. 437-452, Athens, Greece.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J. 2013a. Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems, 26.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J. 2013b. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

Mohammed, S. N. and Gündüç, S. 2022. Degree-based random walk approach for graph embedding. Turkish Journal of Electrical Engineering and Computer Sciences, 30(5): 1868-1881.

Mohammed, S. N. and Gündüç, S. 2023. Tpm: Transition probability matrix - graph structural feature based embedding. Kybernetika, 59(2): 234-253.

Moschopoulos, C., Likothanassis, S., Stamatopoulos, V. and Kossida, S. 2009. Applying graph theory on protein-protein interaction data, In: 16th International Conference on Systems, Signals and Image Processing, IEEE, pp. 1-4.

Nandi, G. and Das, A. 2014. Online social network mining: Current trends and research issues. International Journal of Research in Engineering and Technology, 3(4): 346-350.

Neville, J. and Jensen, D. 2000. Iterative classification in relational data, In: Workshop on Learning Statistical Models from Relational Data, pp. 13-20.

Nguyen, D. and Malliaros, F. D. 2018. Biasedwalk: Biased sampling for representation learning on graphs, In: IEEE International Conference on Big Data, IEEE, pp. 4045-4053.

Nguyen, D. Q., Billingsley, R., Du, L. and Johnson, M. 2015. Improving topic models with latent feature word representations. Transactions of the Association for Computational Linguistics, 3: 299-313.

Ou, M., Cui, P., Pei, J., Zhang, Z. and Zhu, W. 2016. Asymmetric transitivity preserving graph embedding, In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1105-1114.

Palumbo, E., Rizzo, G. and Troncy, R. 2017. Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation, In: Proceedings of the Eleventh ACM Conference on Recommender Systems, pp. 32-36.

Pennington, J., Socher, R. and Manning, C. D. 2014. Glove: Global vectors for word representation, In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1532-1543.

Perozzi, B., Al-Rfou, R. and Skiena, S. 2014. Deepwalk: Online learning of social representations, In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701-710.

Perozzi, B., Kulkarni, V., Chen, H. and Skiena, S. 2017. Don't walk, skip! Online learning of multi-scale network embeddings, In: International Conference on Advances in Social Networks Analysis and Mining, pp. 258-265.

Ribeiro, L., Saverese, P. and Figueiredo, D. 2017. Struc2vec: Learning node representations from structural identity, In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 385-394.

Rossi, R. and Ahmed, N. 2015. The network data repository with interactive graph analytics and visualization, In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 4292-4293.

Roweis, S. and Saul, L. 2000. Nonlinear dimensionality reduction by locally linear embedding. Science, 290(5500): 2323-2326.

Rozemberczki, B., Allen, C. and Sarkar, R. 2021. Multi-scale attributed node embedding. Journal of Complex Networks, 9(2): cnab014.

Salamat, A., Luo, X. and Jafari, A. 2020. Balnode2vec: Balanced random walk based versatile feature learning for networks, In: International Joint Conference on Neural Networks, IEEE, pp. 1-8.

Sammut, C. and Webb, G. I. 2011. Encyclopedia of machine learning, Springer, 1030 pages, New York.

Savaş, S. 2022. Analysis of digital participation with text mining method and 2017 turkey social media referendum. Acta Infologica, 6(1): 65-81.

Savaş, S. ve Topaloğlu, N. 2015. Sosyal medya verileri üzerinden siber İstihbarat faaliyetleri, In: 8. Uluslararası Bilgi Güvenliği ve Kriptoloji Konferansı, pp. 62-68, Ankara.

Savaş, S. ve Topaloğlu, N. 2016. Siber güvenlikte yeni bir boyut: Sosyal medya istihbaratı, In: 18. Akademik Bilişim Konferansı, Aydın.

Savaş, S. ve Topaloğlu, N. 2018. R programlama dili ile sosyal medya analizi: Örnek olay İncelemesi, In: 20. Akademik Bilişim Konferansı, pp. 41-52, Karabük.

Seker, S. E., 2008. Web sitesi. https://bilgisayarkavramlari.com/2008/08/01/graf-sekil-graph/. Erişim Tarihi: 12.01.2023.

Singer, U., Guy, I. and Radinsky, K. 2019. Node embedding over temporal graphs. arXiv preprint arXiv:1903.08889.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y. and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank, In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1631-1642.

Stieglitz, S. and Dang-Xuan, L. 2013. Emotions and information diffusion in social media sentiment of microblogs and sharing behavior. Journal of Management Information Systems, 29(4): 217-248.

Sun, G. and Zhang, X. 2017. Graph embedding with rich information through heterogeneous network. arXiv preprint arXiv:1710.06879 .

Tang, J., Liu, J., Zhang, M. and Mei, Q. 2016. Visualizing large-scale and high-dimensional data, In: Proceedings of the 25th International Conference on World Wide Web, pp. 287-297.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J. and Mei, Q. 2015. Line: Large-scale information network embedding, In: Proceedings of the 24th International Conference on World Wide Web, pp. 1067-1077.

Tenenbaum, J. B., De Silva, V. and Langford, J. C. 2000. A global geometric framework for nonlinear dimensionality reduction. Science, 290(5500): 2319-2323.

Tu, C., Zhang, W., Liu, Z. and Sun, M. 2016. Max-margin deepwalk: Discriminative learning of network representation, In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, pp. 3889-3895.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P. and Bengio, Y. 2017. Graph attention networks. arXiv preprint arXiv:1710.10903.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P. and Bottou, L. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of Machine Learning Research, 11(12): 3371-3408.

Waikhom, L. and Patgiri, R. 2021. Graph neural networks: Methods, applications, and opportunities. arXiv preprint arXiv:2108.10733.

Wang, D., Cui, P. and Zhu, W. 2016. Structural deep network embedding, In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1225-1234.

Wang, H. and Leskovec, J. 2020. Unifying graph convolutional neural networks and label propagation. arXiv preprint arXiv:2002.06755.

Wang, H., Wang, J., Wang, J., Zhao, M., Zhang, W., Zhang, F., Li, W., Xie, X. and Guo, M. 2019. Learning graph representation with generative adversarial nets. IEEE Transactions on Knowledge and Data Engineering, 33(8): 3090-3103.

Wang, X., Bo, D., Shi, C., Fan, S., Ye, Y. and Philip, S. Y. 2022. A survey on heterogeneous graph embedding: Methods, techniques, applications and sources. IEEE Transactions on Big Data, 9(2): 415-436.

Wang, Y., Richard, R. and Mcdonald, D. 2020. Competitive analysis with graph embedding on patent networks, In: 22nd Conference on Business Informatics, pp. 10-19.

Weston, J., Bordes, A., Chopra, S., Rush, A. M., Van Merriënboer, B., Joulin, A. and Mikolov, T. 2016. Towards ai-complete question answering: A set of prerequisite toy tasks, In: 4th International Conference on Learning Representations.

Xia, M., Wang, J. and He, Y. 2013. Brainnet viewer: A network visualization tool for human brain connectomics. PloS One, 8(7): 68910.

Xie, C., Wen, X., Meng, F. and Pang, H. 2023. Random walk graph auto-encoders with ensemble networks in graph embedding. IEEE Access, 11: 55204-55211.

Xin, R. S., Gonzalez, J. E., Franklin, M. J. and Stoica, I. 2013. Graphx: A resilient distributed graph system on spark, In: First International Workshop on Graph Data Management Experiences and Systems, pp. 1-6.

Yacouby, R. and Axman, D. 2020. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models, In: Proceedings of the First workshop on Evaluation and Comparison of NLP Systems, pp. 79-91.

Yang, C., Liu, Z., Zhao, D., Sun, M. and Chang, E. Y. 2015. Network representation learning with rich text information, In: Proceedings of IJCAI, pp. 2111-2117.

Yin, H., Benson, A. R., Leskovec, J. and Gleich, D. F. 2017. Local higher-order graph clustering, In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 555-564.

Yu, Y., Wang, Y., Luo, Z., Yang, Y., Wang, X., Gao, T. and Yu, Q. 2022. Link prediction algorithm based on clustering coefficient and node centrality. Journal of Tsinghua University - Science and Technology, 62(1): 98-104.

Zhang, D., Yin, J., Zhu, X. and Zhang, C. 2016a. Collective classification via discriminative matrix factorization on sparsely labeled networks, In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 1563-1572.

Zhang, D., Yin, J., Zhu, X. and Zhang, C. 2016b. Tline: Scalable transductive network embedding, In: Information Retrieval Technology: 12th Asia Information Retrieval Societies Conference, Springer, pp. 98-110, Beijing, China.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. and Sun, M. 2020. Graph neural networks: A review of methods and applications. AI Open, 1: 57-81.

Zhou, J., Liu, L., Wei, W. and Fan, J. 2022. Network representation learning: From preprocessing, feature extraction to node embedding. ACM Computing Surveys (CSUR), 55(2): 1-35.

Zhu, X., Ghahramani, Z. and Lafferty, J. D. 2003. Semi-supervised learning using gaussian fields and harmonic functions, In: Proceedings of the 20th International Conference on Machine Learning, pp. 912-919.

Zou, W. Y., Socher, R., Cer, D. and Manning, C. D. 2013. Bilingual word embeddings for phrase-based machine translation, In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1393-1398.

ÖZGEÇMİŞ

Kişisel Bilgiler

Adı ve Soyadı : Sarah Abdulkareem Ahmed AHMED

Eğitim

Yüksek Lisans	Çankırı Karatekin Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Mühendisliği Anabilim Dalı	2021-2023
Lisans	Imam Ja'afer Al-sadiq Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü	2011-2015

İş Deneyimi

Yıl	**Kurum**	**Görev**
2015-2018	Kerkük Üniversitesi Fen Dergisi	Editör Sekreteri
2011-2015	IRAK Kuzey Elektrik Dağıtım AŞ.	Mühendis

Akademik Aktiviteler

1. AHMED, S. A. A., SAVAŞ, S., and MOHAMMED, S. N. 2022. Öznitelik Tabanlı Özellik Öğrenme Yöntemiyle Ağ Temsilinin Öğrenilmesi. Proceedings of the 1st International Conference on Innovative Academic Studies (ICIAS), 1681-1687.